//Prototype code for a Planck-scale simulation of a Hubble-volume-sized universe.

//Unfinished original inspiration of Lammps World

//Software was created in partnership with the unofficial Agapepolis science group.

// Copyright 2023 Alexander Vislykh

//

// Licensed under the Apache License, Version 2.0 (the "License");

// you may not use this file except in compliance with the License.

// You may obtain a copy of the License at

//

// http://www.apache.org/licenses/LICENSE-2.0

//

// Unless required by applicable law or agreed to in writing, software

// distributed under the License is distributed on an "AS IS" BASIS,

// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

// See the License for the specific language governing permissions and

// limitations under the License.

var aaseq;

var updated;

var allparticles;

var altitude;

var countb1;

var countb2;

var countb3;

var countcg1;

var countcg2;

var countcg3;

var countl1;

var countl2;

var countl3;

var countcell1;

var countcell2;

var countcell3;

var repeat\_endcell1;

var repeat\_endcell2;

var repeat\_endcell3;

var repeat\_endcg1;

var repeat\_endcg2;

var repeat\_endcg3;

var countsub1;

var countsub2;

var countsub3;

var anchorx;

var anchory;

var anchorz;

var aphid\_genome;

var ant\_genome;

var areached;

var atom\_speed\_x;

var atom\_speed\_y;

var atom\_speed\_z;

var connectome\_window\_1;

var connectome\_window\_2;

var atx;

var aty;

var atz;

var axial\_tilt\_rotation\_coordinate;

var azgenvar;

var bacillus\_genome;

var bee\_genome;

var bonus;

var bool;

var boolyesno;

var box;

var box2;

var brain1;

var brain2;

var brainx;

var brainy;

var brainz;

var browserused;

var buchnera\_genome;

var bvar;

var childnodelist;

var childnodefive;

var childnonefour;

var childnodeone;

var childnodethree;

var childnodetwo;

var cedex;

var cedey;

var cedez;

var cells\_list;

var centerpoint\_x;

var centerpoint\_y;

var centerpoint\_z;

var c1;

var c2;

var c3;

var c4;

var c5;

var c6;

var c7;

var ctring;

var cursor\_eta;

var cursor1x;

var cursor1y;

var cursor1z;

var cursor2x;

var cursor2y;

var cursor2z;

var cursor3x;

var cursor3y;

var cursor3z;

var cursor4x;

var cursor4y;

var cursor4z;

var cursor5x;

var cursor5y;

var cursor5z;

var cursor\_phi;

var cylindrical\_length\_coordinate;

var cylindrical\_radius;

var density\_parameter\_x;

var density\_parameter\_y;

var density\_parameter\_z;

var density\_paramter\_z;

var density\_two;

var dist1;

var dist2;

var dist3;

var dist4;

var dist5;

var dna;

var ductsize;

var eara;

var earb;

var earth\_speed\_\_x;

var earth\_speed\_\_y;

var earth\_speed\_\_z;

var energy;

var eukaryotic\_genomes;

var evar;

var evogen;

var evolength;

var evolink;

var evolinput;

var evolutionary\_genomes;

var evotaxon;

var evoturn;

var evowin;

var exempt;

var flyans;

var flyb;

var flyc1;

var flyc2;

var flyc3;

var flyinput;

var galaxy\_generation\_parameter;

var galaxy\_location\_;

var galaxy\_neighbour\_\_location;

var galaxy\_neighbour\_\_x;

var galaxy\_neighbour\_\_y;

var galaxy\_neighbour\_\_z;

var galaxy\_neighbouring\_\_\_selected\_;

var galaxy\_selected\_\_entity;

var galaxy\_speed\_;

var galaxy\_speed\_\_x;

var galaxy\_speed\_\_y;

var galaxy\_speed\_\_z;

var galaxy\_x\_\_\_default\_;

var galaxy\_y\_\_\_default\_;

var galaxy\_z\_\_\_default\_;

var galaxydistance\_parameter\_\_default;

var galaxylist\_parameter\_\_1;

var gena;

var genb;

var genbank\_window;

var genderiv;

var genlist1;

var genlist2;

var generation\_parameter\_3;

var generation\_parameter\_x;

var generation\_parameter\_y;

var generation\_parameter\_z;

var generation\_parx;

var generation\_pary;

var generation\_par\_z;

var generator\_x;

var generator\_y;

var generator\_z;

var geninsert;

var genome;

var genomelist;

var genomeprevious;

var genomesequence;

var genometwo;

var genomobtained;

var genomeobtained;

var genoutput;

var genraw;

var genrepeat1;

var genrepeat2;

var gentarget;

var genvchloro;

var genvmitsac;

var genvpaul;

var genvpauvchromo;

var grass\_genome;

var guider\_x;

var guider\_y;

var guider\_z;

var hadron;

var hadronbonus;

var hadroncount;

var hadronlist;

var hadronnum;

var holderx;

var holdery;

var holderz;

var hrefforuse;

var hreftwo;

var human\_genome;

var inputcon;

var intextnum;

var isityesno;

var istwogenome;

var l1;

var l2rgen;

var largest1;

var largest2;

var lichen\_cyanobiont\_genome;

var lichen\_symbiont\_genome;

var lipid1;

var lipid2;

var list;

var list\_\_universe\_string\_entities\_\_\_default;

var list\_cells;

var list\_galaxy\_entities\_\_\_default\_;

var list\_stellar\_objects\_\_galactic;

var list\_molecules;

var list\_tissues;

var list\_universes;

var load;

var loc1;

var loc2;

var loc3;

var loc4;

var loc5;

var loc6;

var lunar\_eta\_coordinate;

var lunar\_phi\_coordinate;

var lunar\_rotation\_direction;

var made;

var make;

var material\_temperature\_\_kelvins;

var mitochondrial\_genome;

var molecular\_cloud\_generation\_parameter;

var molsizex;

var molsizey;

var molsizez;

var moon\_layer\_generation\_parameter;

var moon\_mare\_1\_x;

var moon\_mare\_1\_y;

var moon\_mare\_1\_z;

var moon\_mare\_2\_x;

var moon\_mare\_2\_y;

var moon\_mare\_2\_z;

var moon\_mare\_3\_x;

var moon\_mare\_3\_y;

var moon\_mare\_3\_z;

var moon\_outer\_diameter;

var moon\_speed\_x;

var moon\_speed\_y;

var moon\_speed\_z;

var moon\_x;

var moon\_y;

var moon\_z;

var mouseans1;

var mouseans2;

var mouseb;

var mousec1;

var mousec2;

var mousec3;

var mouseinput;

var mouselist;

var mousen1;

var mousen2;

var mousen3;

var mousen4;

var mousen5;

var mousesize;

var mousesizetar;

var mousevar;

var movespeed;

var movex;

var movey;

var movez;

var muscle;

var ncbi\_window;

var nebula\_atom\_randomizer;

var nebula\_generation\_parameter;

var nebula\_size\_;

var nebula\_x\_;

var nebula\_y\_;

var nebula\_z\_;

var nematode\_genome;

var nerve;

var nerve2;

var nerve3;

var neuron1;

var neuron2;

var newline;

var nfirst;

var nodelistextra;

var nodelistfive;

var nodelistfour;

var nodelistthree;

var nodelisttwo;

var nsec;

var nucleoid\_size\_x;

var nucleoid\_size\_y;

var nucleoid\_size\_z;

var nullbrane1;

var nullbrane2;

var nullbrane3;

var num1;

var num2;

var num3;

var num4;

var num5;

var par1;

var par2;

var par3;

var par4;

var par5;

var par6;

var par7;

var par8;

var par9;

var par10;

var particle1;

var particle2;

var pelagibacter\_genome;

var peptidesequence;

var peptideacidlist;

var permit;

var placeholder\_eta;

var plant\_genomes;

var planet\_randomizer;

var planet\_x;

var planet\_y;

var planet\_z;

var plasma\_randomization\_parameter;

var ploidy;

var ploidynum;

var ploidyprevious;

var pointclick;

var primary\_parameter\_\_universal\_\_main\_;

var prokaryotic\_genomes;

var randomizer;

var relative;

var relative\_x;

var relative\_y;

var relative\_z;

var repeat\_x;

var repeat\_y;

var repeat\_z;

var rho;

var rnaseq;

var rotation\_distance\_moon;

var rotation\_speed;

var rotational\_speed\_vector;

var rotational\_speed\_x;

var rotational\_speed\_y;

var rotational\_speed\_z;

var rotifer\_genome;

var shrub\_genome;

var selstring1;

var selstring2;

var seta;

var sizeend;

var sizemax;

var sizemid;

var sizemin;

var sizestart;

var sphi;

var spincom;

var spineta;

var spinphi;

var spinrelx;

var spinrely;

var spinrelz;

var spinrho;

var spirlength;

var spirnum;

var spirulina\_genome;

var spirwidth;

var solar\_core\_\_generation\_parameter;

var solargeneration\_parameter\_;

var solarneighbour;

var sorter;

var spawnerx;

var spawnery;

var spawnerz;

var spawnerx2;

var spawnery2;

var spawnerz2;

var spruce\_genome;

var srho;

var star\_selected\_;

var star\_speed\_;

var star\_speed\_\_x;

var star\_speed\_\_y;

var star\_speed\_\_z;

var starter\_x;

var starter\_y;

var starter\_z;

var stellar\_distance\_\_parameter;

var stellarparameter\_1\_\_default\_;

var string\_x;

var string\_y;

var string\_z;

var string\_solar\_distance\_\_\_selected\_;

var string\_solar\_distance\_\_length\_parameter;

var string\_solar\_\_distance\_parameter;

var stwidth;

var sun\_x;

var sun\_y;

var sun\_z;

var sun\_neighbour\_x\_;

var sun\_neighbour\_\_y;

var sun\_neighbour\_\_z;

var symvolgen;

var tardigrade\_genome;

var target1;

var target2;

var textloc;

var textpiece;

var thickabs;

var thickabs2;

var thickbod;

var thickbod2;

var thickrel;

var thickrel2;

var tilted\_rotation\_inclination;

var tilted\_rotation\_radius;

var tilted\_rotation\_z;

var tilted\_speed\_x;

var tilted\_speed\_y;

var tilted\_speed\_z;

var tissues\_list;

var topographical\_window;

var topoinput;

var topology\_download\_parameter;

var topoloutput;

var tree\_genome;

var treebox;

var treeinput;

var universe;

var unused\_variable;

var unused\_window;

var used\_density\_parameter;

var used\_density\_parameter\_\_lunar\_;

var used\_density\_value\_\_solar\_;

var used\_diameter\_\_lunar\_;

var used\_diameter\_\_solar\_;

var used\_link;

var used\_radius\_layer\_earth;

var used\_window\_evol;

var used\_window\_two;

var vcen1;

var vcen2;

var vcen3;

var vloc;

var vp1;

var vp2;

var viable;

var viable2;

var wavechange;

var widthmax;

var window\_connectome\_1;

var window\_connectome\_2;

var window\_connectome\_3;

var yesno;

const Blob = require('node-blob');

main();

function Generate\_neutron() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

hadronbonus = 0;

cursor1x = cursor1x + 600000000000000000;

addupquark();

cursor1x = cursor1x + 30000000000000000000;

adddownquark();

cursor1x = cursor1x - 15000000000000000000;

cursor1y = cursor1x + 30000000000000000000;

hadronbonus = 1;

adddownquark();

cursor1y = cursor1x - 30000000000000000000;

cursor1x = cursor1x - 15000000000000000000;

cursor1x = cursor1x - 600000000000000000;

}

}

function math\_random\_int(a, b) {

if (a > b) {

// Swap a and b to ensure a is smaller.

var c = a;

a = b;

b = c;

}

return Math.floor(Math.random() \* (b - a + 1) + a);

}

function Generate\_system() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

cursor1x = sun\_x - 8.61780266e+42;

cursor1y = sun\_y - 8.61780266e+42;

cursor1z = sun\_z - 8.61780266e+42;

material\_temperature\_\_kelvins = 10000000;

solar\_core\_\_generation\_parameter = Math.round((8.61780266e+42 \* 2) / 2.012891816e+24);

for (var count = 0; count < solar\_core\_\_generation\_parameter; count++) {

cursor1x = cursor1x + 2.012891816e+24;

for (var count2 = 0; count2 < solar\_core\_\_generation\_parameter; count2++) {

cursor1y = cursor1y + 2.012891816e+24;

for (var count3 = 0; count3 < nebula\_generation\_parameter; count3++) {

cursor1z = cursor1z + 2.012891816e+24;

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) <= 8.61780266e+42) {

randomize\_sun\_plasma();

}

}

cursor1z = cursor1z - solar\_core\_\_generation\_parameter \* 2.012891816e+24;

}

cursor1y = cursor1y - solar\_core\_\_generation\_parameter \* 2.012891816e+24;

}

cursor1x = cursor1x - solar\_core\_\_generation\_parameter \* 2.012891816e+24;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

cursor1x = sun\_x - 4.305807369e+43;

cursor1y = sun\_y - 4.305807369e+43;

cursor1z = sun\_z - 4.305807369e+43;

material\_temperature\_\_kelvins = 1000000;

solar\_core\_\_generation\_parameter = Math.round((4.305807369e+43 \* 2) / 5.831076748e+25);

for (var count6 = 0; count6 < solar\_core\_\_generation\_parameter; count6++) {

cursor1x = cursor1x + 5.831076748e+25;

for (var count5 = 0; count5 < solar\_core\_\_generation\_parameter; count5++) {

cursor1y = cursor1y + 5.831076748e+25;

for (var count4 = 0; count4 < nebula\_generation\_parameter; count4++) {

cursor1z = cursor1z + 5.831076748e+25;

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) <= 4.305807369e+43) {

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) > 3.016230933e+43) {

randomize\_sun\_plasma();

}

}

}

cursor1z = cursor1z - solar\_core\_\_generation\_parameter \* 5.831076748e+25;

}

cursor1y = cursor1y - solar\_core\_\_generation\_parameter \* 5.831076748e+25;

}

cursor1x = cursor1x - solar\_core\_\_generation\_parameter \* 5.831076748e+25;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_solar\_ = 3.016230933e+43;

used\_density\_value\_\_solar\_ = 5.644204962e+24;

cursor1x = sun\_x - used\_diameter\_\_solar\_;

cursor1y = sun\_y - used\_diameter\_\_solar\_;

cursor1z = sun\_z - used\_diameter\_\_solar\_;

material\_temperature\_\_kelvins = 4500000;

solar\_core\_\_generation\_parameter = Math.round((used\_diameter\_\_solar\_ \* 2) / used\_density\_value\_\_solar\_);

for (var count9 = 0; count9 < solar\_core\_\_generation\_parameter; count9++) {

cursor1x = cursor1x + used\_density\_value\_\_solar\_;

for (var count8 = 0; count8 < solar\_core\_\_generation\_parameter; count8++) {

cursor1y = cursor1y + used\_density\_value\_\_solar\_;

for (var count7 = 0; count7 < nebula\_generation\_parameter; count7++) {

cursor1z = cursor1z + used\_density\_value\_\_solar\_;

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) <= 3.016230933e+43) {

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) > 8.61780266e+42) {

randomize\_sun\_plasma();

}

}

}

cursor1z = cursor1z - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1y = cursor1y - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1x = cursor1x - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_solar\_ = 4.308901333e+43;

used\_density\_value\_\_solar\_ = 7.873190804e+27;

cursor1x = sun\_x - used\_diameter\_\_solar\_;

cursor1y = sun\_y - used\_diameter\_\_solar\_;

cursor1z = sun\_z - used\_diameter\_\_solar\_;

material\_temperature\_\_kelvins = 5900;

solar\_core\_\_generation\_parameter = Math.round((used\_diameter\_\_solar\_ \* 2) / used\_density\_value\_\_solar\_);

for (var count12 = 0; count12 < solar\_core\_\_generation\_parameter; count12++) {

cursor1x = cursor1x + used\_density\_value\_\_solar\_;

for (var count11 = 0; count11 < solar\_core\_\_generation\_parameter; count11++) {

cursor1y = cursor1y + used\_density\_value\_\_solar\_;

for (var count10 = 0; count10 < nebula\_generation\_parameter; count10++) {

cursor1z = cursor1z + used\_density\_value\_\_solar\_;

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) <= 4.308901333e+43) {

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) > 4.305807369e+43) {

randomize\_sun\_plasma();

}

}

}

cursor1z = cursor1z - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1y = cursor1y - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1x = cursor1x - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_solar\_ = 9e+45;

used\_density\_value\_\_solar\_ = 3.5e+30;

cursor1x = sun\_x - used\_diameter\_\_solar\_;

cursor1y = sun\_y - used\_diameter\_\_solar\_;

cursor1z = sun\_z - used\_diameter\_\_solar\_;

material\_temperature\_\_kelvins = 293;

solar\_core\_\_generation\_parameter = Math.round((used\_diameter\_\_solar\_ \* 2) / used\_density\_value\_\_solar\_);

for (var count15 = 0; count15 < solar\_core\_\_generation\_parameter; count15++) {

cursor1x = cursor1x + used\_density\_value\_\_solar\_;

for (var count14 = 0; count14 < solar\_core\_\_generation\_parameter; count14++) {

cursor1y = cursor1y + used\_density\_value\_\_solar\_;

for (var count13 = 0; count13 < nebula\_generation\_parameter; count13++) {

cursor1z = cursor1z + used\_density\_value\_\_solar\_;

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) > 4.308901333e+43) {

if (Math.sqrt((Math.pow(Math.abs(sun\_y - cursor1y), 2) + Math.pow(Math.abs(sun\_x - cursor1x), 2)) + Math.pow(Math.abs(sun\_z - cursor1z), 2)) < 9e+45) {

add\_randomized\_solar\_photon();

}

}

}

cursor1z = cursor1z - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1y = cursor1y - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

cursor1x = cursor1x - solar\_core\_\_generation\_parameter \* used\_density\_value\_\_solar\_;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

planet\_randomizer = math\_random\_int(1, 100);

if (planet\_randomizer <= 30) {

generate\_planet();

}

}

}

function generate() {

primary\_parameter\_\_universal\_\_main\_ = true;

list\_molecules = [];

list\_tissues = [];

cells\_list = [];

list\_galaxy\_entities\_\_\_default\_ = [];

galaxylist\_parameter\_\_1 = 0;

for (var count17 = 0; count17 < 2000000000000; count17++) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

galaxy\_generation\_parameter = true;

list\_galaxy\_entities\_\_\_default\_.push([[math\_random\_int(4.6e+56, 5.5e+61), math\_random\_int(4.6e+56, 5.5e+61), math\_random\_int(4.6e+56, 5.5e+61)], [math\_random\_int(0, 2100000) / 1079000000, math\_random\_int(0, 2100000) / 1079000000, math\_random\_int(0, 2100000) / 1079000000]]);

galaxy\_selected\_\_entity = list\_galaxy\_entities\_\_\_default\_[(galaxylist\_parameter\_\_1 + 1) - 1];

galaxylist\_parameter\_\_1 = galaxylist\_parameter\_\_1 + 1;

galaxy\_location\_ = galaxy\_selected\_\_entity[0];

galaxy\_speed\_ = galaxy\_selected\_\_entity[1];

galaxy\_x\_\_\_default\_ = galaxy\_location\_[0];

galaxy\_y\_\_\_default\_ = galaxy\_location\_[1];

galaxy\_z\_\_\_default\_ = galaxy\_location\_[2];

galaxy\_speed\_\_x = galaxy\_speed\_[0];

galaxy\_speed\_\_y = galaxy\_speed\_[1];

galaxy\_speed\_\_z = galaxy\_speed\_[2];

galaxydistance\_parameter\_\_default = 0;

}

if (list\_galaxy\_entities\_\_\_default\_.length >= 2) {

var repeat\_end = list\_galaxy\_entities\_\_\_default\_.length - 1;

for (var count16 = 0; count16 < repeat\_end; count16++) {

galaxy\_neighbouring\_\_\_selected\_ = list\_galaxy\_entities\_\_\_default\_[(galaxydistance\_parameter\_\_default + 1) - 1];

galaxydistance\_parameter\_\_default = galaxydistance\_parameter\_\_default + 1;

galaxy\_neighbour\_\_location = galaxy\_neighbouring\_\_\_selected\_[0];

galaxy\_neighbour\_\_x = galaxy\_neighbouring\_\_\_selected\_[0];

galaxy\_neighbour\_\_y = galaxy\_neighbouring\_\_\_selected\_[1];

galaxy\_neighbour\_\_z = galaxy\_neighbouring\_\_\_selected\_[2];

if (Math.abs(galaxy\_x\_\_\_default\_ - galaxy\_neighbour\_\_x) < 6e+55) {

galaxy\_generation\_parameter = false;

} else {

if (Math.abs(galaxy\_y\_\_\_default\_ - galaxy\_neighbour\_\_y) < 6e+55) {

galaxy\_generation\_parameter = false;

} else {

if (Math.abs(galaxy\_z\_\_\_default\_ - galaxy\_neighbour\_\_z) < 6e+55) {

galaxy\_generation\_parameter = false;

}

}

}

}

}

if (galaxy\_generation\_parameter == true) {

generate\_2();

}

}

}

function generate\_2() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

cursor1x = galaxy\_x\_\_\_default\_;

cursor1y = galaxy\_y\_\_\_default\_;

cursor1z = galaxy\_z\_\_\_default\_;

centerpoint\_x = galaxy\_x\_\_\_default\_;

centerpoint\_y = galaxy\_y\_\_\_default\_;

centerpoint\_z = galaxy\_z\_\_\_default\_;

material\_temperature\_\_kelvins = 10;

atom\_speed\_x = galaxy\_speed\_\_x;

atom\_speed\_y = galaxy\_speed\_\_y;

atom\_speed\_z = galaxy\_speed\_\_z;

for (var count20 = 0; count20 < 100000000000000000000; count20++) {

cursor1x = cursor1x + 6.18e+21;

for (var count19 = 0; count19 < 100000000000000000000; count19++) {

cursor1y = cursor1y + 6.18e+21;

var repeat\_end2 = 4.75e+23;

for (var count18 = 0; count18 < repeat\_end2; count18++) {

cursor1z = cursor1z + 6.18e+21;

Generate\_neutron();

}

cursor1z = cursor1z - 4.75e+23 \* 6.18e+21;

}

cursor1y = cursor1y - 100000000000000000000 \* 6.18e+21;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

cursor1x = galaxy\_x\_\_\_default\_;

cursor1y = galaxy\_y\_\_\_default\_;

cursor1z = galaxy\_z\_\_\_default\_;

cursor1x = galaxy\_x\_\_\_default\_ - 1.4e+55;

cursor1y = galaxy\_y\_\_\_default\_ - 1.4e+55;

cursor1z = galaxy\_y\_\_\_default\_ - 1.4e+55;

molecular\_cloud\_generation\_parameter = Math.round(2.8e+55 / 1.3e+34);

for (var count23 = 0; count23 < molecular\_cloud\_generation\_parameter; count23++) {

cursor1x = cursor1x + 1.9e+34;

for (var count22 = 0; count22 < molecular\_cloud\_generation\_parameter; count22++) {

cursor1y = cursor1y + 1.9e+34;

for (var count21 = 0; count21 < molecular\_cloud\_generation\_parameter; count21++) {

cursor1z = cursor1z + 1.9e+34;

if (Math.abs(cursor1y - galaxy\_y\_\_\_default\_) <= 5e+53) {

if (Math.sqrt(Math.pow(Math.abs(galaxy\_x\_\_\_default\_ - cursor1x), 2) + Math.pow(Math.abs(galaxy\_z\_\_\_default\_ - cursor1z), 2)) <= 1.4e+55) {

if (Math.sqrt(Math.pow(Math.abs(galaxy\_x\_\_\_default\_ - cursor1x), 2) + Math.pow(Math.abs(galaxy\_z\_\_\_default\_ - cursor1z), 2)) >= 6e+54) {

randomize\_nebula\_atom();

}

}

}

}

cursor1z = cursor1z - molecular\_cloud\_generation\_parameter \* 1.9e+34;

}

cursor1y = cursor1y - molecular\_cloud\_generation\_parameter \* 1.9e+34;

}

for (var count27 = 0; count27 < 1000; count27++) {

nebula\_x\_ = galaxy\_x\_\_\_default\_ + math\_random\_int(-4e+55, 4e+55);

nebula\_y\_ = galaxy\_y\_\_\_default\_ + math\_random\_int(-4e+53, 4e+53);

nebula\_z\_ = galaxy\_z\_\_\_default\_ + math\_random\_int(-4e+55, 4e+55);

cursor1x = nebula\_x\_;

cursor1y = nebula\_y\_;

cursor1z = nebula\_z\_;

nebula\_size\_ = math\_random\_int(5e+50, 1.7e+53);

nebula\_generation\_parameter = Math.round(nebula\_size\_ / 1.2e+32);

if (Math.abs(nebula\_x\_ - galaxy\_x\_\_\_default\_) >= 1.522e+55) {

if (Math.abs(nebula\_z\_ - galaxy\_z\_\_\_default\_) >= 1.522e+55) {

cursor1x = cursor1x - nebula\_size\_ / 2;

cursor1y = cursor1y - nebula\_size\_ / 2;

cursor1z = cursor1z - nebula\_size\_ / 2;

hollow\_nebula();

for (var count26 = 0; count26 < nebula\_generation\_parameter; count26++) {

cursor1x = cursor1x + 1.2e+32;

for (var count25 = 0; count25 < nebula\_generation\_parameter; count25++) {

cursor1y = cursor1y + 1.2e+32;

for (var count24 = 0; count24 < nebula\_generation\_parameter; count24++) {

cursor1z = cursor1z + 1.2e+32;

if (Math.sqrt((Math.pow(Math.abs(nebula\_y\_ - cursor1y), 2) + Math.pow(Math.abs(nebula\_x\_ - cursor1x), 2)) + Math.pow(Math.abs(nebula\_z\_ - cursor1z), 2)) <= nebula\_size\_ / 2) {

randomize\_nebula\_atom();

}

}

cursor1z = cursor1z - nebula\_generation\_parameter \* 1.2e+32;

}

cursor1y = cursor1y - nebula\_generation\_parameter \* 1.2e+32;

}

}

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

list\_stellar\_objects\_\_galactic = [];

stellarparameter\_1\_\_default\_ = 0;

for (var count30 = 0; count30 < 250000000000; count30++) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

stellar\_distance\_\_parameter = 0;

list\_stellar\_objects\_\_galactic.push([galaxy\_x\_\_\_default\_ + math\_random\_int(-5.8e+55, 5.8e+55), galaxy\_y\_\_\_default\_ + math\_random\_int(-2e+54, -2e+54), galaxy\_z\_\_\_default\_ + math\_random\_int(-5.8e+55, 5.8e+55), [0, 0, 0]]);

star\_selected\_ = list\_stellar\_objects\_\_galactic.slice(-1)[0];

sun\_x = star\_selected\_[0];

sun\_y = star\_selected\_[1];

sun\_z = star\_selected\_[2];

solargeneration\_parameter\_ = true;

if (list\_stellar\_objects\_\_galactic.length >= 2) {

var repeat\_end3 = list\_stellar\_objects\_\_galactic.length;

for (var count28 = 0; count28 < repeat\_end3; count28++) {

solarneighbour = list\_stellar\_objects\_\_galactic[(stellar\_distance\_\_parameter + 1) - 1];

stellar\_distance\_\_parameter = stellar\_distance\_\_parameter + 1;

sun\_neighbour\_x\_ = solarneighbour[0];

sun\_neighbour\_\_y = solarneighbour[1];

sun\_neighbour\_\_z = solarneighbour[2];

if (Math.abs(sun\_x - sun\_neighbour\_x\_) < 3e+48) {

solargeneration\_parameter\_ = false;

} else if (Math.abs(sun\_y - sun\_neighbour\_\_y) < 3e+48) {

solargeneration\_parameter\_ = false;

} else if (Math.abs(sun\_z - sun\_neighbour\_\_z) < 3e+48) {

solargeneration\_parameter\_ = false;

}

}

stellar\_distance\_\_parameter = 0;

}

if (Math.abs(sun\_x - galaxy\_x\_\_\_default\_) < 5e+50) {

solargeneration\_parameter\_ = false;

} else if (Math.abs(sun\_y - galaxy\_y\_\_\_default\_) < 5e+50) {

solargeneration\_parameter\_ = false;

} else if (Math.abs(sun\_z - galaxy\_z\_\_\_default\_) < 5e+50) {

solargeneration\_parameter\_ = false;

}

if (solargeneration\_parameter\_ == true) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end4 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count29 = 0; count29 < repeat\_end4; count29++) {

if (string\_solar\_\_distance\_parameter + 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (Math.sqrt((Math.pow(Math.abs(string\_y - sun\_y), 2) + Math.pow(Math.abs(string\_x - string\_x), 2)) + Math.pow(Math.abs(string\_z - sun\_z), 2)) < 2.7e+48) {

list\_\_universe\_string\_entities\_\_\_default.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

star\_speed\_ = star\_selected\_[3];

if (sun\_x > galaxy\_x\_\_\_default\_) {

star\_speed\_.splice(2, 0, galaxy\_speed\_\_z + 0.00015);

} else {

star\_speed\_.splice(2, 0, galaxy\_speed\_\_z - 0.00015);

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

list.splice(1, 0, galaxy\_speed\_\_y);

}

if (sun\_z > galaxy\_z\_\_\_default\_) {

star\_speed\_.splice(0, 0, galaxy\_speed\_\_x + 0.00015);

} else {

star\_speed\_.splice(0, 0, galaxy\_speed\_\_x - 0.00015);

}

star\_speed\_\_x = star\_speed\_[0];

star\_speed\_\_y = star\_speed\_[1];

star\_speed\_\_z = star\_speed\_[2];

}

if (solargeneration\_parameter\_ == true) {

Generate\_system();

}

}

}

}

}

function generate\_planet() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

planet\_y = sun\_y + 8.49566147e+45;

planet\_z = sun\_z - 3.676401009e+45;

planet\_x = sun\_x;

earth\_speed\_\_x = star\_speed\_\_x + 0.00009933538755001;

earth\_speed\_\_y = star\_speed\_\_y;

earth\_speed\_\_z = star\_speed\_\_z;

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 7.555459298e+40;

used\_density\_parameter = 1.486387933e+25;

material\_temperature\_\_kelvins = 5475;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

for (var count33 = 0; count33 < generation\_parameter\_3; count33++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count32 = 0; count32 < generation\_parameter\_3; count32++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count31 = 0; count31 < generation\_parameter\_3; count31++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 7.555459298e+40) {

set\_earth\_rotation();

randomize\_earth\_core\_alloy\_atom();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 7.555459298e+40;

used\_density\_parameter = 1.486387933e+25;

material\_temperature\_\_kelvins = 5475;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

cursor1x = cursor1x - used\_density\_parameter / 2;

cursor1y = cursor1y - used\_density\_parameter / 2;

cursor1z = cursor1z - used\_density\_parameter / 2;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

for (var count36 = 0; count36 < generation\_parameter\_3; count36++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count35 = 0; count35 < generation\_parameter\_3; count35++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count34 = 0; count34 < generation\_parameter\_3; count34++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 7.555459298e+40) {

set\_earth\_rotation();

randomize\_earth\_core\_alloy\_atom();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 2.156492682e+41;

used\_density\_parameter = 1.248140589e+25;

material\_temperature\_\_kelvins = 4273.15;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

for (var count39 = 0; count39 < generation\_parameter\_3; count39++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count38 = 0; count38 < generation\_parameter\_3; count38++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count37 = 0; count37 < generation\_parameter\_3; count37++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 7.555459298e+40) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

set\_dynamo\_metal\_movement();

randomize\_earth\_core\_alloy\_atom();

}

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 3.818569969e+41;

density\_parameter\_x = 1.237670034e+25;

density\_parameter\_y = 1.237670034e+25;

density\_paramter\_z = 1.237670034e+25;

material\_temperature\_\_kelvins = 2400;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_x = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_y);

generation\_parameter\_z = Math.round((used\_radius\_layer\_earth \* 2) / density\_paramter\_z);

for (var count42 = 0; count42 < generation\_parameter\_x; count42++) {

cursor1x = cursor1x + density\_parameter\_x;

for (var count41 = 0; count41 < generation\_parameter\_y; count41++) {

cursor1y = cursor1y + density\_parameter\_y;

for (var count40 = 0; count40 < generation\_parameter\_z; count40++) {

cursor1z = cursor1z + density\_paramter\_z;

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 2.156492682e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_paramter\_z;

}

cursor1y = cursor1y - generation\_parameter\_y \* density\_parameter\_y;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 3.933046626e+41;

density\_parameter\_x = 1.36170366e+25;

density\_parameter\_y = 1.36170366e+25;

density\_paramter\_z = 1.36170366e+25;

material\_temperature\_\_kelvins = 1580 + math\_random\_int(-1000, 1000) / 1000;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_x = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_y);

generation\_parameter\_z = Math.round((used\_radius\_layer\_earth \* 2) / density\_paramter\_z);

for (var count45 = 0; count45 < generation\_parameter\_x; count45++) {

cursor1x = cursor1x + density\_parameter\_x;

for (var count44 = 0; count44 < generation\_parameter\_y; count44++) {

cursor1y = cursor1y + density\_parameter\_y;

for (var count43 = 0; count43 < generation\_parameter\_z; count43++) {

cursor1z = cursor1z + density\_paramter\_z;

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1y - planet\_y;

relative\_z = cursor1z - planet\_z;

if (Math.abs(relative\_z) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_y) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_x) < 5) {

relative\_x = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_y / relative\_x);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + (Math.pow(Math.abs(relative\_y), 2) + Math.pow(Math.abs(relative\_z), 2)))));

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (lunar\_eta\_coordinate > Math.PI \* 1.5) {

if (lunar\_eta\_coordinate < Math.PI \* 1.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > Math.PI) {

if (lunar\_eta\_coordinate < Math.PI + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_phi\_coordinate > 1.570794) {

if (lunar\_phi\_coordinate < 1.570796) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > Math.PI / 2) {

if (lunar\_eta\_coordinate < Math.PI \* 0.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > 0) {

if (lunar\_eta\_coordinate < 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.818569969e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 3.880449243e+41) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_paramter\_z;

}

cursor1y = cursor1y - generation\_parameter\_y \* density\_parameter\_y;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_radius\_layer\_earth = 3.933046626e+41;

density\_parameter\_x =1.438361028e+25 ;

density\_parameter\_y =1.438361028e+25 ;

density\_paramter\_z = 1.438361028e+25;

material\_temperature\_\_kelvins = 500;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_x = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((used\_radius\_layer\_earth \* 2) / density\_parameter\_y);

generation\_parameter\_z = Math.round((used\_radius\_layer\_earth \* 2) / density\_paramter\_z);

for (var count48 = 0; count48 < generation\_parameter\_x; count48++) {

cursor1x = cursor1x + density\_parameter\_x;

for (var count47 = 0; count47 < generation\_parameter\_y; count47++) {

cursor1y = cursor1y + density\_parameter\_y;

for (var count46 = 0; count46 < generation\_parameter\_z; count46++) {

cursor1z = cursor1z + density\_paramter\_z;

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1y - planet\_y;

relative\_z = cursor1z - planet\_z;

if (Math.abs(relative\_z) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_y) < 5) {

relative\_y = 6;

}

if (Math.abs(relative\_x) < 5) {

relative\_x = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_y / relative\_x);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + (Math.pow(Math.abs(relative\_y), 2) + Math.pow(Math.abs(relative\_z), 2)))));

if (lunar\_phi\_coordinate > 1.570794) {

if (lunar\_phi\_coordinate > 1.570796) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (lunar\_eta\_coordinate > Math.PI \* 1.5) {

if (lunar\_eta\_coordinate > Math.PI \* 1.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > Math.PI) {

if (lunar\_eta\_coordinate > Math.PI + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (primary\_parameter\_\_universal\_\_main\_ == false) {

unused\_variable = 5;

} else if (lunar\_eta\_coordinate > Math.PI / 2) {

if (lunar\_eta\_coordinate > Math.PI \* 0.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > 0) {

if (lunar\_eta\_coordinate > 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

}

}

}

} else {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (lunar\_eta\_coordinate > Math.PI \* 1.5) {

if (lunar\_eta\_coordinate > Math.PI \* 1.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > Math.PI) {

if (lunar\_eta\_coordinate > Math.PI + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (primary\_parameter\_\_universal\_\_main\_ == false) {

unused\_variable = 5;

} else if (lunar\_eta\_coordinate > Math.PI / 2) {

if (lunar\_eta\_coordinate > Math.PI \* 0.5 + 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

} else if (lunar\_eta\_coordinate > 0) {

if (lunar\_eta\_coordinate > 0.000002) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 3.880449243e+41) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_magma\_atom();

}

}

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_paramter\_z;

}

cursor1y = cursor1y - generation\_parameter\_y \* density\_parameter\_y;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

unused\_variable = 5;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

terrain();

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

moon\_y = planet\_y + 2.072269171e+43;

moon\_x = planet\_x;

moon\_z = planet\_z - 1.167663668e+43;

moon\_speed\_x = earth\_speed\_\_x + 0.0000031274;

moon\_speed\_z = earth\_speed\_\_z + 2.816e-7;

moon\_speed\_y = earth\_speed\_\_y;

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.485102e+40;

used\_density\_parameter\_\_lunar\_ = 1.767272052e+25;

material\_temperature\_\_kelvins = 1600;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count51 = 0; count51 < moon\_layer\_generation\_parameter; count51++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count50 = 0; count50 < moon\_layer\_generation\_parameter; count50++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count49 = 0; count49 < moon\_layer\_generation\_parameter; count49++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.485102e+23) {

set\_moon\_rotation();

iron\_metallic\_();

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.485102e+40;

used\_density\_parameter\_\_lunar\_ = 1.767272052e+25;

material\_temperature\_\_kelvins = 1600;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x - used\_density\_parameter\_\_lunar\_ / 2;

cursor1y = cursor1y - used\_density\_parameter\_\_lunar\_ / 2;

cursor1z = cursor1z - used\_density\_parameter\_\_lunar\_ / 2;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count54 = 0; count54 < moon\_layer\_generation\_parameter; count54++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count53 = 0; count53 < moon\_layer\_generation\_parameter; count53++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count52 = 0; count52 < moon\_layer\_generation\_parameter; count52++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.485102e+23) {

set\_moon\_rotation();

iron\_metallic\_();

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 2.042016026e+40;

used\_density\_parameter\_\_lunar\_ = 1.637946288e+25;

material\_temperature\_\_kelvins = 1700;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count57 = 0; count57 < moon\_layer\_generation\_parameter; count57++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count56 = 0; count56 < moon\_layer\_generation\_parameter; count56++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count55 = 0; count55 < moon\_layer\_generation\_parameter; count55++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.485102e+23) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 2.042016026e+40) {

set\_moon\_rotation();

iron\_metallic\_();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count60 = 0; count60 < moon\_layer\_generation\_parameter; count60++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count59 = 0; count59 < moon\_layer\_generation\_parameter; count59++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count58 = 0; count58 < moon\_layer\_generation\_parameter; count58++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + 0;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 2;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 2;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count63 = 0; count63 < moon\_layer\_generation\_parameter; count63++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count62 = 0; count62 < moon\_layer\_generation\_parameter; count62++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count61 = 0; count61 < moon\_layer\_generation\_parameter; count61++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 2;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 2;

cursor1z = cursor1z + 0;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count66 = 0; count66 < moon\_layer\_generation\_parameter; count66++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count65 = 0; count65 < moon\_layer\_generation\_parameter; count65++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count64 = 0; count64 < moon\_layer\_generation\_parameter; count64++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 2;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 1;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 2;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count69 = 0; count69 < moon\_layer\_generation\_parameter; count69++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count68 = 0; count68 < moon\_layer\_generation\_parameter; count68++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count67 = 0; count67 < moon\_layer\_generation\_parameter; count67++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 4;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 4;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 4;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count72 = 0; count72 < moon\_layer\_generation\_parameter; count72++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count71 = 0; count71 < moon\_layer\_generation\_parameter; count71++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count70 = 0; count70 < moon\_layer\_generation\_parameter; count70++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 4;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.75;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count75 = 0; count75 < moon\_layer\_generation\_parameter; count75++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count74 = 0; count74 < moon\_layer\_generation\_parameter; count74++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count73 = 0; count73 < moon\_layer\_generation\_parameter; count73++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.25;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count78 = 0; count78 < moon\_layer\_generation\_parameter; count78++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count77 = 0; count77 < moon\_layer\_generation\_parameter; count77++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count76 = 0; count76 < moon\_layer\_generation\_parameter; count76++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.043965223e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.25;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.75;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count81 = 0; count81 < moon\_layer\_generation\_parameter; count81++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count80 = 0; count80 < moon\_layer\_generation\_parameter; count80++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count79 = 0; count79 < moon\_layer\_generation\_parameter; count79++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 2.042016026e+40) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.043965223e+41) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

moon\_mare\_1\_x = moon\_x - 1e+41;

moon\_mare\_1\_y = moon\_y + 1.5e+40;

moon\_mare\_1\_z = moon\_z + 5e+40;

moon\_mare\_2\_x = moon\_x - 1e+41;

moon\_mare\_2\_y = moon\_y + 7e+40;

moon\_mare\_2\_z = moon\_z + 1e+40;

moon\_mare\_3\_x = moon\_x - 1e+41;

moon\_mare\_3\_y = moon\_y + 7e+40;

moon\_mare\_3\_z = moon\_z - 4.5e+40;

}

moon\_outer\_diameter = 1.07490486e+41;

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = moon\_outer\_diameter;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count84 = 0; count84 < moon\_layer\_generation\_parameter; count84++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count83 = 0; count83 < moon\_layer\_generation\_parameter; count83++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count82 = 0; count82 < moon\_layer\_generation\_parameter; count82++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + 0;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 2;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 2;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count87 = 0; count87 < moon\_layer\_generation\_parameter; count87++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count86 = 0; count86 < moon\_layer\_generation\_parameter; count86++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count85 = 0; count85 < moon\_layer\_generation\_parameter; count85++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 2;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 2;

cursor1z = cursor1z + 0;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count90 = 0; count90 < moon\_layer\_generation\_parameter; count90++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count89 = 0; count89 < moon\_layer\_generation\_parameter; count89++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count88 = 0; count88 < moon\_layer\_generation\_parameter; count88++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 2;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 1;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 2;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count93 = 0; count93 < moon\_layer\_generation\_parameter; count93++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count92 = 0; count92 < moon\_layer\_generation\_parameter; count92++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count91 = 0; count91 < moon\_layer\_generation\_parameter; count91++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_4();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ / 4;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 4;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ / 4;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count96 = 0; count96 < moon\_layer\_generation\_parameter; count96++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count95 = 0; count95 < moon\_layer\_generation\_parameter; count95++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count94 = 0; count94 < moon\_layer\_generation\_parameter; count94++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ / 4;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.75;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count99 = 0; count99 < moon\_layer\_generation\_parameter; count99++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count98 = 0; count98 < moon\_layer\_generation\_parameter; count98++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count97 = 0; count97 < moon\_layer\_generation\_parameter; count97++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = moon\_outer\_diameter;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 1075;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.25;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count102 = 0; count102 < moon\_layer\_generation\_parameter; count102++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count101 = 0; count101 < moon\_layer\_generation\_parameter; count101++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count100 = 0; count100 < moon\_layer\_generation\_parameter; count100++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

diamond\_3();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.07490486e+41;

used\_density\_parameter\_\_lunar\_ = 2.207233687e+25;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_ \* 0.25;

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_ \* 0.75;

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_ \* 0.75;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count105 = 0; count105 < moon\_layer\_generation\_parameter; count105++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count104 = 0; count104 < moon\_layer\_generation\_parameter; count104++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count103 = 0; count103 < moon\_layer\_generation\_parameter; count103++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > 1.043965223e+41) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < moon\_outer\_diameter) {

if (Math.sqrt((Math.pow(Math.abs(moon\_mare\_1\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_mare\_1\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_mare\_1\_z - cursor1z), 2)) < moon\_outer\_diameter) {

set\_moon\_rotation();

iron\_unbound\_();

} else if (Math.sqrt((Math.pow(Math.abs(moon\_mare\_2\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_mare\_2\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_mare\_2\_z - cursor1z), 2)) < 4.5e+40) {

set\_moon\_rotation();

iron\_unbound\_();

} else if (Math.sqrt((Math.pow(Math.abs(moon\_mare\_3\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_mare\_3\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_mare\_3\_z - cursor1z), 2)) < 6e+40) {

set\_moon\_rotation();

iron\_unbound\_();

} else {

set\_moon\_rotation();

diamond\_3();

}

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

used\_diameter\_\_lunar\_ = 1.09490486e+41;

used\_density\_parameter\_\_lunar\_ = 8e+27;

material\_temperature\_\_kelvins = 295;

cursor1x = moon\_x - used\_diameter\_\_lunar\_;

cursor1y = moon\_y - used\_diameter\_\_lunar\_;

cursor1z = moon\_z - used\_diameter\_\_lunar\_;

moon\_layer\_generation\_parameter = Math.round((used\_diameter\_\_lunar\_ \* 2) / used\_density\_parameter\_\_lunar\_);

for (var count108 = 0; count108 < moon\_layer\_generation\_parameter; count108++) {

cursor1x = cursor1x + used\_density\_parameter\_\_lunar\_;

for (var count107 = 0; count107 < moon\_layer\_generation\_parameter; count107++) {

cursor1y = cursor1y + used\_density\_parameter\_\_lunar\_;

for (var count106 = 0; count106 < moon\_layer\_generation\_parameter; count106++) {

cursor1z = cursor1z + used\_density\_parameter\_\_lunar\_;

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) > moon\_outer\_diameter) {

if (Math.sqrt((Math.pow(Math.abs(moon\_y - cursor1y), 2) + Math.pow(Math.abs(moon\_x - cursor1x), 2)) + Math.pow(Math.abs(moon\_z - cursor1z), 2)) < 1.09490486e+41) {

set\_moon\_rotation();

lunar\_helium();

}

}

}

cursor1z = cursor1z - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

cursor1y = cursor1y - moon\_layer\_generation\_parameter \* used\_density\_parameter\_\_lunar\_;

}

}

}

}

}

}

function generate\_proton() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

hadronbonus = 0;

cursor1x = cursor1x + 600000000000000000;

addupquark();

cursor1x = cursor1x + 30000000000000000000;

addupquark();

cursor1x = cursor1x - 15000000000000000000;

cursor1y = cursor1x + 30000000000000000000;

adddownquark();

cursor1y = cursor1x - 30000000000000000000;

cursor1x = cursor1x - 15000000000000000000;

cursor1x = cursor1x - 600000000000000000;

}

}

function main() {

newline = String.fromCharCode(10);

topology\_download\_parameter = false;

downloadsx();

list\_universes = [];

var repeat\_end201 = 1.79767e+308;

for (var count201 = 0; count201 < repeat\_end201; count201++) {

list\_universes.push([math\_random\_int(1, 10000000000000000000000000000000000000000000000000), []]);

list\_\_universe\_string\_entities\_\_\_default = (list\_universes.slice(-1)[0])[1];

generate();

}

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

physical\_change();

setInterval(physical\_change, 1000);

}

function randomize\_earth\_core\_alloy\_atom() {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 8000) {

iron\_metallic\_();

} else {

nickel\_metallic\_();

}

}

function randomize\_magma\_atom() {

randomizer = math\_random\_int(1, 1000000000);

if (randomizer <= 700) {

my\_1\_rhenium();

} else if (randomizer <= 1700) {

my\_2\_rhodium();

} else if (randomizer <= 2700) {

my\_3\_iridium();

} else if (randomizer <= 3700) {

my\_4\_ruthenium();

} else if (randomizer <= 4700) {

my\_5\_tellurium();

} else if (randomizer <= 6200) {

my\_6\_osmium();

} else if (randomizer <= 10200) {

my\_7\_gold();

} else if (randomizer <= 15200) {

my\_8\_platinum();

} else if (randomizer <= 23700) {

my\_9\_bismuth();

} else if (randomizer <= 38700) {

my\_10\_palladium();

} else if (randomizer <= 88700) {

my\_11\_selenium();

} else if (randomizer <= 100700) {

my\_12\_silver();

} else if (randomizer <= 163700) {

my\_13\_mercury();

} else if (randomizer <= 248700) {

my\_14\_cadmium();

} else if (randomizer <= 398700) {

my\_15\_antimony();

} else if (randomizer <= 598700) {

my\_16\_indium();

} else if (randomizer <= 598700) {

my\_17\_iodine();

} else if (randomizer <= 848700) {

my\_18\_thulium();

} else if (randomizer <= 1298700) {

my\_19\_lutetium();

} else if (randomizer <= 1818700) {

my\_20\_thallium();

} else if (randomizer <= 2618700) {

my\_21\_terbium();

} else {

randomize\_magma\_atom2();

}

}

function randomize\_magma\_atom2() {

if (randomizer <= 3500000) {

my\_22\_holmium();

} else if (randomizer <= 4700000) {

my\_23\_molybdenum();

} else if (randomizer <= 6000000) {

my\_24\_tungsten();

} else if (randomizer <= 7200000) {

my\_25\_germanium();

} else if (randomizer <= 7450000) {

my\_26\_tantalum();

} else if (randomizer <= 8950000) {

my\_27\_arsenic();

} else if (randomizer <= 10950000) {

my\_28\_europium();

} else if (randomizer <= 13750000) {

my\_29\_tin();

} else if (randomizer <= 15750000) {

my\_30\_bromine();

} else if (randomizer <= 18050000) {

my\_31\_uranium();

} else if (randomizer <= 20450000) {

my\_32\_beryllium();

} else if (randomizer <= 23150000) {

my\_33\_caesium();

} else if (randomizer <= 25950000) {

my\_34\_hafnium();

} else if (randomizer <= 28950000) {

my\_35\_ytterbium();

} else if (randomizer <= 31000000) {

my\_36\_erbium();

} else if (randomizer <= 34200000) {

my\_37\_dysprosium();

} else if (randomizer <= 37700000) {

my\_38\_gadolinium();

} else if (randomizer <= 42900000) {

my\_39\_samarium();

} else if (randomizer <= 49100000) {

my\_40\_praseodimium();

} else if (randomizer <= 56100000) {

my\_41\_thorium();

} else if (randomizer <= 65300000) {

my\_42\_boron();

} else {

randomize\_magma\_atom3();

}

}

function randomize\_magma\_atom3() {

if (randomizer <= 74900000) {

my\_43\_lead();

} else if (randomizer <= 84900000) {

my\_44\_gallium();

} else if (randomizer <= 98900000) {

my\_45\_niobium();

} else if (randomizer <= 117900000) {

my\_46\_lithium();

} else if (randomizer <= 137900000) {

my\_47\_scandium();

} else if (randomizer <= 157900000) {

my\_48\_cobalt();

} else if (randomizer <= 179900000) {

my\_49\_nitrogen();

} else if (randomizer <= 204900000) {

my\_50\_yttrium();

} else if (randomizer <= 223900000) {

my\_51\_lanthanum();

} else if (randomizer <= 256900000) {

my\_52\_neodymium();

} else if (randomizer <= 295900000) {

my\_53\_cerium();

} else if (randomizer <= 335900000) {

my\_54\_copper();

} else if (randomizer <= 395900000) {

my\_55\_zinc();

} else if (randomizer <= 455900000) {

my\_56\_nickel();

} else if (randomizer <= 525900000) {

my\_57\_rubidium();

} else if (randomizer <= 610000000) {

my\_58\_chromium();

} else if (randomizer <= 700000000) {

my\_59\_vanadium();

} else if (randomizer <= 802000000) {

my\_60\_chlorine();

} else if (randomizer <= 922000000) {

my\_61\_zirconium();

} else if (randomizer <= 1067000000) {

my\_62\_carbon();

} else if (randomizer <= 1232000000) {

my\_62\_carbon();

} else {

randomize\_magma\_atom4();

}

}

function randomize\_magma\_atom4() {

if (randomizer <= 1432000000) {

my\_62\_carbon();

} else if (randomizer <= 1782000000) {

my\_62\_carbon();

} else if (randomizer <= 2152000000) {

my\_63\_sulphur();

} else if (randomizer <= 2477000000) {

my\_64\_strontium();

} else if (randomizer <= 2977000000) {

my\_65\_barium();

} else if (randomizer <= 4000000000) {

my\_66\_fluorine();

} else if (randomizer <= 5050000000) {

my\_67\_manganese();

} else if (randomizer <= 6450000000) {

my\_68\_phosphorus();

} else if (randomizer <= 12550000000) {

my\_69\_hydrogen();

} else if (randomizer <= 33450000000) {

my\_70\_titanium();

} else if (randomizer <= 56750000000) {

my\_71\_potassium();

} else if (randomizer <= 80350000000) {

my\_72\_magnesium();

} else if (randomizer <= 121850000000) {

my\_73\_sodium();

} else if (randomizer <= 178150000000) {

my\_74\_calcium();

} else if (randomizer <= 260450000000) {

my\_75\_iron();

} else if (randomizer <= 350450000000) {

my\_76\_aluminium();

} else if (randomizer <= 542450000000) {

my\_77\_silicon();

} else {

my\_78\_oxygen();

}

}

function randomize\_nebula\_atom() {

nebula\_atom\_randomizer = math\_random\_int(1, 1e+30);

if (primary\_parameter\_\_universal\_\_main\_ == true) {

if (cursor1x > galaxy\_x\_\_\_default\_) {

atom\_speed\_z = galaxy\_speed\_\_z + 0.00015;

} else {

atom\_speed\_z = galaxy\_speed\_\_z - 0.00015;

}

if (primary\_parameter\_\_universal\_\_main\_ == true) {

atom\_speed\_y = galaxy\_speed\_\_y;

}

if (cursor1z > galaxy\_z\_\_\_default\_) {

atom\_speed\_x = galaxy\_speed\_\_x + 0.00015;

} else {

atom\_speed\_x = galaxy\_speed\_\_x - 0.00015;

}

}

if (nebula\_atom\_randomizer < 7.3e+29) {

air\_hydrogen();

} else if (nebula\_atom\_randomizer < 9.8e+29) {

helium\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.88e+29) {

oxygen\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.916e+29) {

carbon\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.932e+29) {

iron\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.933e+29) {

iron\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.944e+29) {

neon\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.953e+29) {

nitrogen\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.96e+29) {

silicon\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.965e+29) {

magnesium\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.969e+29) {

sulphur\_unbound\_();

} else if (nebula\_atom\_randomizer < 9.96907e+29) {

phosphorus\_unbound\_();

} else {

argon\_unbound\_();

}

}

function randomize\_sun\_plasma() {

if (primary\_parameter\_\_universal\_\_main\_ == true) {

atom\_speed\_x = star\_speed\_\_x;

atom\_speed\_y = star\_speed\_\_y;

atom\_speed\_z = star\_speed\_\_z;

}

plasma\_randomization\_parameter = math\_random\_int(1, 10000);

if (plasma\_randomization\_parameter <= 625) {

helium\_unbound\_();

} else {

my\_69\_hydrogen();

}

}

function set\_dynamo\_metal\_movement() {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) > 9.555459298e+40) {

if (Math.sqrt((Math.pow(Math.abs(planet\_y - cursor1y), 2) + Math.pow(Math.abs(planet\_x - cursor1x), 2)) + Math.pow(Math.abs(planet\_z - cursor1z), 2)) < 2e+41) {

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1y - planet\_y;

relative\_z = cursor1z - planet\_z;

if (Math.abs(relative\_z) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_y) < 5) {

relative\_y = 6;

}

if (Math.abs(relative\_x) < 5) {

relative\_x = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_y / relative\_x);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + (Math.pow(Math.abs(relative\_y), 2) + Math.pow(Math.abs(relative\_z), 2)))));

lunar\_rotation\_direction = lunar\_eta\_coordinate + 0.5 \* Math.PI;

rotation\_distance\_moon = Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + Math.pow(Math.abs(relative\_z), 2)));

if (Math.abs(rotation\_distance\_moon) < 5) {

rotation\_distance\_moon = 6;

}

rotational\_speed\_vector = rotation\_distance\_moon / 2e+41;

rotation\_speed = rotational\_speed\_vector \* 1.5473667749e-12;

rotational\_speed\_x = rotation\_speed \* (Math.cos(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

rotational\_speed\_y = rotation\_speed \* (Math.sin(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

atom\_speed\_x = rotational\_speed\_x + atom\_speed\_x;

atom\_speed\_z = atom\_speed\_z;

atom\_speed\_y = rotational\_speed\_y + atom\_speed\_y;

}

}

}

function set\_earth\_rotation() {

relative\_x = cursor1x - moon\_x;

relative\_y = cursor1y - moon\_y;

relative\_z = cursor1z - moon\_z;

if (Math.abs(relative\_z) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_y) < 5) {

relative\_y = 6;

}

if (Math.abs(relative\_x) < 5) {

relative\_x = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_y / relative\_x);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + (Math.pow(Math.abs(relative\_y), 2) + Math.pow(Math.abs(relative\_z), 2)))));

lunar\_rotation\_direction = lunar\_eta\_coordinate + 0.5 \* Math.PI;

rotation\_distance\_moon = Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + Math.pow(Math.abs(relative\_y), 2)));

if (Math.abs(rotation\_distance\_moon) < 5) {

rotation\_distance\_moon = 6;

}

rotational\_speed\_vector = rotation\_distance\_moon / 3.942328517e+41;

rotation\_speed = rotational\_speed\_vector \* 0.000001551406606767;

rotational\_speed\_x = rotation\_speed \* (Math.cos(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

rotational\_speed\_y = rotation\_speed \* (Math.sin(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

atom\_speed\_x = rotational\_speed\_x + moon\_speed\_x;

atom\_speed\_z = moon\_speed\_z;

atom\_speed\_y = rotational\_speed\_y + moon\_speed\_y;

}

function set\_moon\_rotation() {

relative\_x = cursor1x - moon\_x;

relative\_y = cursor1y - moon\_y;

relative\_z = cursor1z - moon\_z;

if (Math.abs(relative\_z) < 5) {

relative\_z = 6;

}

if (Math.abs(relative\_y) < 5) {

relative\_y = 6;

}

if (Math.abs(relative\_x) < 5) {

relative\_x = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_y / relative\_x);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + (Math.pow(Math.abs(relative\_y), 2) + Math.pow(Math.abs(relative\_z), 2)))));

lunar\_rotation\_direction = lunar\_eta\_coordinate + 0.5 \* Math.PI;

rotation\_distance\_moon = Math.sqrt(Math.abs(Math.pow(Math.abs(relative\_x), 2) + Math.pow(Math.abs(relative\_y), 2)));

if (Math.abs(rotation\_distance\_moon) < 5) {

rotation\_distance\_moon = 6;

}

rotational\_speed\_vector = rotation\_distance\_moon / 1.07490486e+41;

rotation\_speed = rotational\_speed\_vector \* 1.5473667749e-8;

rotational\_speed\_x = rotation\_speed \* (Math.cos(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

rotational\_speed\_y = rotation\_speed \* (Math.sin(lunar\_rotation\_direction) \* Math.sin(Math.PI / 2));

atom\_speed\_x = rotational\_speed\_x + moon\_speed\_x;

atom\_speed\_z = moon\_speed\_z;

atom\_speed\_y = rotational\_speed\_y + moon\_speed\_y;

}

function hollow\_nebula() {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end200 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count200 = 0; count200 < repeat\_end200; count200++) {

if (string\_solar\_\_distance\_parameter + 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (Math.sqrt((Math.pow(Math.abs(string\_y - nebula\_y\_), 2) + Math.pow(Math.abs(string\_x - nebula\_x\_), 2)) + Math.pow(Math.abs(string\_z - nebula\_z\_), 2)) < nebula\_size\_ / 2) {

list\_\_universe\_string\_entities\_\_\_default.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

function downloadsx() {

unused\_variable = 5;

spirnum = [];

prokaryotic\_genomes = [];

eukaryotic\_genomes = [];

plant\_genomes = [];

evolutionary\_genomes = [];

symvolgen = [];

utilized\_genomes();

genomelists();

evolutionary\_genomes2();

freshwater\_genomes();

marine\_genomes();

additional\_genomes();

symbiogenetic\_genomes();

fly\_connectome();

vertebrate\_connectome();

eukaryotic\_genomes.unshift(prokaryotic\_genomes[math\_random\_int(1, prokaryotic\_genomes.length) - 1]);

eukaryotic\_genomes.unshift(prokaryotic\_genomes[math\_random\_int(1, prokaryotic\_genomes.length) - 1]);

plant\_genomes.unshift(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

plant\_genomes.unshift(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

}

async function utilized\_genomes() {

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/375/235/GCA\_902375235.1\_Morex\_v1.0\_update\_x/GCA\_902375235.1\_Morex\_v1.0\_update\_x\_genomic.fna.gz");

grass\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/114/115/GCF\_002114115.1\_ASM211411v1/GCF\_002114115.1\_ASM211411v1\_genomic.fna.gz");

tree\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/515/GCF\_000004515.5\_Glycine\_max\_v2.1/GCF\_000004515.5\_Glycine\_max\_v2.1\_genomic.fna.gz");

shrub\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/067/695/GCA\_900067695.1\_Pabies01/GCA\_900067695.1\_Pabies01\_genomic.fna.gz");

spruce\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/015/582/655/GCF\_015582655.1\_ASM1558265v1/GCF\_015582655.1\_ASM1558265v1\_genomic.fna.gz");

bacillus\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/012/932/795/GCF\_012932795.1\_ASM1293279v1/GCF\_012932795.1\_ASM1293279v1\_genomic.fna.gz");

pelagibacter\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/210/375/GCF\_000210375.1\_ASM21037v1/GCF\_000210375.1\_ASM21037v1\_genomic.fna.gz");

spirulina\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/254/395/GCF\_003254395.2\_Amel\_HAv3.1/GCF\_003254395.2\_Amel\_HAv3.1\_genomic.fna.gz");

bee\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/651/465/GCF\_003651465.1\_ASM365146v1/GCF\_003651465.1\_ASM365146v1\_genomic.fna.gz");

ant\_genome = await load.blob();

load = await fetch("ftp://ftp.ncbi.nlm.nih.gov/refseq/H\_sapiens/annotation/GRCh38\_latest/refseq\_identifiers/GRCh38\_latest\_genomic.fna.gz");

human\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/005/508/785/GCF\_005508785.1\_pea\_aphid\_22Mar2018\_4r6ur/GCF\_005508785.1\_pea\_aphid\_22Mar2018\_4r6ur\_genomic.fna.gz");

aphid\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/352/455/GCA\_006352455.1\_Btisc\_TiscarSM28/GCA\_006352455.1\_Btisc\_TiscarSM28\_genomic.fna.gz");

rotifer\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/020/025/GCF\_000020025.1\_ASM2002v1/GCF\_000020025.1\_ASM2002v1\_genomic.fna.gz");

lichen\_cyanobiont\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/118/135/GCA\_002118135.1\_TrTZW2008\_1.0/GCA\_002118135.1\_TrTZW2008\_1.0\_genomic.fna.gz");

lichen\_symbiont\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/731/375/GCA\_009731375.1\_PmicroMYN1\_1.0/GCA\_009731375.1\_PmicroMYN1\_1.0\_genomic.fna.gz");

genvpaul = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/refseq/release/mitochondrion/mitochondrion.1.1.genomic.fna.gz");

mitochondrial\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/985/GCF\_000002985.6\_WBcel235/GCF\_000002985.6\_WBcel235\_genomic.fna.gz");

nematode\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/082/055/GCA\_002082055.1\_nHd\_3.1/GCA\_002082055.1\_nHd\_3.1\_genomic.fna.gz");

tardigrade\_genome = await load.blob();

load = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/009/605/GCF\_000009605.1\_ASM960v1/GCF\_000009605.1\_ASM960v1\_genomic.fna.gz");

buchnera\_genome = await load.blob();

}

//genome downloading, reading, and unzipping algorithms are currently unfinished

//This version is therefore not yet fully functional

//The unzipping code is not yet included

async function genomelists() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/archaea/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* prokaryotic\_genomes.length);

prokaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

extragen1();

extragen2();

extragen3();

extragen4();

eukaryotic\_genomes2();

}

async function extragen1() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/bacteria/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* prokaryotic\_genomes.length);

prokaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen2() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/viral/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* prokaryotic\_genomes.length);

prokaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen3() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/metagenomes/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* prokaryotic\_genomes.length);

prokaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen4() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/other/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* prokaryotic\_genomes.length);

prokaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function eukaryotic\_genomes2() {

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/fungi/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

extragen5();

extragen6();

extragen7();

extragen8();

extragen9();

plant\_genomes2();

}

async function extragen5() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/invertebrate/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen6() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/protozoa/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen7() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/vertebrate\_mammalian/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen8() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/vertebrate\_other/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function extragen9() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/plant/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* eukaryotic\_genomes.length);

eukaryotic\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function plant\_genomes2() {

//requiring path and fs modules

const path = require('path');

const fs = require('fs');

//joining path of directory

const directoryPath = "https://ftp.ncbi.nih.gov/genomes/genbank/plant/";

//passsing directoryPath and callback function

fs.readdir(directoryPath, function (err, files) {

//handling error

if (err) {

return console.log('Unable to scan directory: ' + err);

}

//listing all files using forEach

files.forEach(async function (file) {

if (path.extname(file) == ".gz") {

load = await fetch(file);

genomobtained = await load.blob();

var tmpX = Math.floor(Math.random() \* plant\_genomes.length);

plant\_genomes.splice(tmpX, 0, genomobtained);

}

});

});

}

async function evolutionary\_genomes2() {

unused\_variable = 5;

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/405/GCF\_000001405.39\_GRCh38.p13/GCF\_000001405.39\_GRCh38.p13\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/880/755/GCF\_002880755.1\_Clint\_PTRv2/GCF\_002880755.1\_Clint\_PTRv2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/013/052/645/GCF\_013052645.1\_Mhudiblu\_PPA\_v0/GCF\_013052645.1\_Mhudiblu\_PPA\_v0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/008/122/165/GCF\_008122165.1\_Kamilah\_GGO\_v0/GCF\_008122165.1\_Kamilah\_GGO\_v0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/880/775/GCF\_002880775.1\_Susie\_PABv2/GCF\_002880775.1\_Susie\_PABv2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/009/828/535/GCF\_009828535.2\_HMol\_V2/GCF\_009828535.2\_HMol\_V2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/006/542/625/GCF\_006542625.1\_Asia\_NLE\_v1/GCF\_006542625.1\_Asia\_NLE\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/025/065/GCA\_004025065.1\_SemEnt\_v1\_BIUU/GCA\_004025065.1\_SemEnt\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/409/795/GCF\_000409795.2\_Chlorocebus\_sabeus\_1.1/GCF\_000409795.2\_Chlorocebus\_sabeus\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/008/728/515/GCF\_008728515.1\_Panubis1.0/GCF\_008728515.1\_Panubis1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/802/615/GCA\_004802615.1\_BGI\_mandrill\_1.0/GCA\_004802615.1\_BGI\_mandrill\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/951/045/GCF\_000951045.1\_Mleu.le\_1.0/GCF\_000951045.1\_Mleu.le\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/955/945/GCF\_000955945.1\_Caty\_1.0/GCF\_000955945.1\_Caty\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/772/465/GCA\_000772465.1\_Charlie1.0/GCA\_000772465.1\_Charlie1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/339/765/GCF\_003339765.1\_Mmul\_10/GCF\_003339765.1\_Mmul\_10\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/118/495/GCA\_003118495.1\_macFus\_1.0/GCA\_003118495.1\_macFus\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/364/345/GCF\_000364345.1\_Macaca\_fascicularis\_5.0/GCF\_000364345.1\_Macaca\_fascicularis\_5.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/026/645/GCA\_004026645.1\_PitPit\_v1\_BIUU/GCA\_004026645.1\_PitPit\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/604/975/GCF\_001604975.1\_Cebus\_imitator-1.0/GCF\_001604975.1\_Cebus\_imitator-1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/235/385/GCF\_000235385.1\_SaiBol1.0/GCF\_000235385.1\_SaiBol1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/952/055/GCF\_000952055.2\_Anan\_2.0/GCF\_000952055.2\_Anan\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/885/GCA\_004024885.1\_SagImp\_v1\_BIUU/GCA\_004024885.1\_SagImp\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/009/663/435/GCF\_009663435.1\_Callithrix\_jacchus\_cj1700\_1.1/GCF\_009663435.1\_Callithrix\_jacchus\_cj1700\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/785/GCA\_004024785.1\_AteGeo\_v1\_BIUU/GCA\_004024785.1\_AteGeo\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/698/545/GCF\_001698545.1\_ASM169854v1/GCF\_001698545.1\_ASM169854v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/007/565/055/GCF\_007565055.1\_ASM756505v1/GCF\_007565055.1\_ASM756505v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/164/805/GCF\_000164805.1\_Tarsius\_syrichta-2.0.1/GCF\_000164805.1\_Tarsius\_syrichta-2.0.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/815/GCA\_004027815.1\_NycCou\_v1\_BIUU/GCA\_004027815.1\_NycCou\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/665/GCA\_004024665.1\_LemCat\_v1\_BIUU/GCA\_004024665.1\_LemCat\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/258/685/GCA\_003258685.1\_Prosim\_1.0/GCA\_003258685.1\_Prosim\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/262/665/GCA\_001262665.1\_Eflavifronsk33QCA/GCA\_001262665.1\_Eflavifronsk33QCA\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/262/655/GCA\_001262655.1\_Emacaco\_refEf\_BWA\_oneround/GCA\_001262655.1\_Emacaco\_refEf\_BWA\_oneround\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/275/GCA\_004027275.1\_EulFul\_v1\_BIUU/GCA\_004027275.1\_EulFul\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/956/105/GCF\_000956105.1\_Pcoq\_1.0/GCF\_000956105.1\_Pcoq\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/363/605/GCA\_004363605.1\_IndInd\_v1\_BIUU/GCA\_004363605.1\_IndInd\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/086/735/GCA\_008086735.1\_ASM808673v1/GCA\_008086735.1\_ASM808673v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/750/955/GCA\_008750955.1\_Mmit\_1.0/GCA\_008750955.1\_Mmit\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/750/915/GCA\_008750915.1\_Mjon\_1.0/GCA\_008750915.1\_Mjon\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/165/445/GCF\_000165445.2\_Mmur\_3.0/GCF\_000165445.2\_Mmur\_3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/181/295/GCF\_000181295.1\_OtoGar3/GCF\_000181295.1\_OtoGar3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/145/GCA\_004027145.1\_DauMad\_v1\_BIUU/GCA\_004027145.1\_DauMad\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/696/425/GCF\_000696425.1\_G\_variegatus-3.0.2/GCF\_000696425.1\_G\_variegatus-3.0.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/365/275/GCA\_004365275.1\_TupTan\_v1\_BIUU/GCA\_004365275.1\_TupTan\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/026/855/GCA\_004026855.1\_LepAme\_v1\_BIUU/GCA\_004026855.1\_LepAme\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/760/805/GCA\_009760805.1\_CIBIO-ISEM\_LeTim\_1.1/GCA\_009760805.1\_CIBIO-ISEM\_LeTim\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/014/633/375/GCF\_014633375.1\_OchPri4.0/GCF\_014633375.1\_OchPri4.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/185/GCA\_004027185.1\_GliGli\_v1\_BIUU/GCA\_004027185.1\_GliGli\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/533/835/GCA\_014533835.1\_Woodchuck\_1.0/GCA\_014533835.1\_Woodchuck\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/458/135/GCF\_001458135.1\_marMar2.1/GCF\_001458135.1\_marMar2.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/547/115/GCA\_006547115.1\_GSC\_porc\_1.0/GCA\_006547115.1\_GSC\_porc\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/015/741/225/GCA\_015741225.1\_Hydrochoerus\_hydrochaeris\_HiC/GCA\_015741225.1\_Hydrochoerus\_hydrochaeris\_HiC\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/276/665/GCF\_000276665.1\_ChiLan1.0/GCF\_000276665.1\_ChiLan1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/247/695/GCF\_000247695.1\_HetGla\_female\_1.0/GCF\_000247695.1\_HetGla\_female\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/686/445/GCA\_902686445.2\_mSciCar1.2/GCA\_902686445.2\_mSciCar1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/316/645/GCA\_011316645.1\_ASM1131664v1/GCA\_011316645.1\_ASM1131664v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/984/765/GCF\_001984765.1\_C.can\_genome\_v1.0/GCF\_001984765.1\_C.can\_genome\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/151/885/GCF\_000151885.1\_Dord\_2.0/GCF\_000151885.1\_Dord\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/280/705/GCF\_000280705.1\_JacJac1.0/GCF\_000280705.1\_JacJac1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/903/992/535/GCF\_903992535.1\_mArvAmp1.1/GCF\_903992535.1\_mArvAmp1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/204/375/GCF\_002204375.1\_MunDraft-v1.0/GCF\_002204375.1\_MunDraft-v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/223/135/GCF\_000223135.1\_CriGri\_1.0/GCF\_000223135.1\_CriGri\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/151/735/GCF\_000151735.1\_Cavpor3.0/GCF\_000151735.1\_Cavpor3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/635/GCF\_000001635.27\_GRCm39/GCF\_000001635.27\_GRCm39\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/895/GCF\_000001895.5\_Rnor\_6.0/GCF\_000001895.5\_Rnor\_6.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/011/064/425/GCF\_011064425.1\_Rrattus\_CSIRO\_v1/GCF\_011064425.1\_Rrattus\_CSIRO\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/363/575/GCA\_004363575.1\_SolPar\_v1\_BIUU/GCA\_004363575.1\_SolPar\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/296/755/GCF\_000296755.1\_EriEur2.0/GCF\_000296755.1\_EriEur2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/260/355/GCF\_000260355.1\_ConCri1.0/GCF\_000260355.1\_ConCri1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/181/275/GCF\_000181275.1\_SorAra2.0/GCF\_000181275.1\_SorAra2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/014/570/555/GCF\_014570555.1\_YNU\_ManPten\_2.0/GCF\_014570555.1\_YNU\_ManPten\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/160/815/GCF\_003160815.1\_VulVul2.2/GCF\_003160815.1\_VulVul2.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/887/905/GCA\_001887905.1\_LycPicSAfr1.0/GCA\_001887905.1\_LycPicSAfr1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/007/922/845/GCA\_007922845.1\_UniMelb\_Wolf\_Refassem\_1/GCA\_007922845.1\_UniMelb\_Wolf\_Refassem\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/285/GCF\_000002285.5\_Dog10K\_Boxer\_Tasha/GCF\_000002285.5\_Dog10K\_Boxer\_Tasha\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/007/445/GCF\_002007445.1\_ASM200744v2/GCF\_002007445.1\_ASM200744v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/584/765/GCF\_003584765.1\_ASM358476v1/GCF\_003584765.1\_ASM358476v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/344/425/GCA\_003344425.1\_ASM334442v1/GCA\_003344425.1\_ASM334442v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/687/225/GCF\_000687225.1\_UrsMar\_1.0/GCF\_000687225.1\_UrsMar\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/348/235/GCF\_004348235.1\_GSC\_HSeal\_1.0/GCF\_004348235.1\_GSC\_HSeal\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/007/465/GCA\_002007465.1\_ASM200746v1/GCA\_002007465.1\_ASM200746v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/015/708/975/GCA\_015708975.1\_Plotor\_platanus500/GCA\_015708975.1\_Plotor\_platanus500\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/697/995/GCA\_003697995.1\_ASM369799v1/GCA\_003697995.1\_ASM369799v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/023/965/GCA\_004023965.1\_SpiGra\_v1\_BIUU/GCA\_004023965.1\_SpiGra\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/655/055/GCA\_902655055.2\_mLutLut1.2/GCA\_902655055.2\_mLutLut1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/288/905/GCF\_002288905.1\_ASM228890v2/GCF\_002288905.1\_ASM228890v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/625/GCA\_004024625.1\_MelCap\_v1\_BIUU/GCA\_004024625.1\_MelCap\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/215/625/GCF\_000215625.1\_MusPutFur1.0/GCF\_000215625.1\_MusPutFur1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/108/605/GCA\_900108605.1\_NNQGG.v01/GCA\_900108605.1\_NNQGG.v01\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/006/375/GCA\_900006375.2\_Gulo\_2.2\_annotated/GCA\_900006375.2\_Gulo\_2.2\_annotated\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/585/GCA\_004024585.1\_ParHer\_v1\_BIUU/GCA\_004024585.1\_ParHer\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/023/785/GCA\_004023785.1\_MunMun\_v1\_BIUU/GCA\_004023785.1\_MunMun\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/006/229/205/GCF\_006229205.1\_meerkat\_22Aug2017\_6uvM2\_HiC/GCF\_006229205.1\_meerkat\_22Aug2017\_6uvM2\_HiC\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/692/635/GCA\_008692635.1\_BGI\_CrCroc\_1.0/GCA\_008692635.1\_BGI\_CrCroc\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/709/585/GCF\_003709585.1\_Aci\_jub\_2/GCF\_003709585.1\_Aci\_jub\_2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/327/715/GCF\_003327715.1\_PumCon1.0/GCF\_003327715.1\_PumCon1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/181/335/GCF\_000181335.3\_Felis\_catus\_9.0/GCF\_000181335.3\_Felis\_catus\_9.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/661/375/GCA\_900661375.1\_LYPA1.0/GCA\_900661375.1\_LYPA1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/464/555/GCF\_000464555.1\_PanTig1.0/GCF\_000464555.1\_PanTig1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/023/805/GCA\_004023805.1\_PanOnc\_v1\_BIUU/GCA\_004023805.1\_PanOnc\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/857/705/GCF\_001857705.1\_PanPar1.0/GCF\_001857705.1\_PanPar1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/795/835/GCA\_008795835.1\_PanLeo1.0/GCA\_008795835.1\_PanLeo1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/729/225/GCA\_902729225.1\_Ma\_sr-lr\_union100/GCA\_902729225.1\_Ma\_sr-lr\_union100\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/115/265/GCF\_004115265.1\_mRhiFer1\_v1.p/GCF\_004115265.1\_mRhiFer1\_v1.p\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/903/992/545/GCA\_903992545.1\_mPipPip1.1/GCA\_903992545.1\_mPipPip1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/151/845/GCF\_000151845.1\_Pvam\_2.0/GCF\_000151845.1\_Pvam\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/863/925/GCF\_002863925.1\_EquCab3.0/GCF\_002863925.1\_EquCab3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/305/755/GCF\_001305755.1\_ASM130575v1/GCF\_001305755.1\_ASM130575v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/696/695/GCF\_000696695.1\_Burgud/GCF\_000696695.1\_Burgud\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/025/025/GCA\_004025025.1\_TapTer\_v1\_BIUU/GCA\_004025025.1\_TapTer\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/634/535/GCA\_013634535.1\_ASM1363453v1/GCA\_013634535.1\_ASM1363453v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/803/125/GCF\_000803125.2\_CamDro3/GCF\_000803125.2\_CamDro3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/767/855/GCF\_000767855.1\_Ca\_bactrianus\_MBC\_1.0/GCF\_000767855.1\_Ca\_bactrianus\_MBC\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/239/585/GCA\_013239585.1\_ASM1323958v1/GCA\_013239585.1\_ASM1323958v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/745/GCA\_004024745.2\_CatWag\_v2\_BIUU\_UCD/GCA\_004024745.2\_CatWag\_v2\_BIUU\_UCD\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/003/025/GCF\_000003025.6\_Sscrofa11.1/GCF\_000003025.6\_Sscrofa11.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/065/GCA\_004027065.2\_HipAmp\_v2\_BIUU\_UCD/GCA\_004027065.2\_HipAmp\_v2\_BIUU\_UCD\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/965/GCA\_004024965.2\_ASM402496v2/GCA\_004024965.2\_ASM402496v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/660/835/GCA\_001660835.1\_ASM166083v1/GCA\_001660835.1\_ASM166083v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/408/565/GCA\_006408565.1\_GFE/GCA\_006408565.1\_GFE\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/515/GCA\_004027515.2\_AntAmePen\_v2\_BIUU\_UCD/GCA\_004027515.2\_AntAmePen\_v2\_BIUU\_UCD\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/102/435/GCF\_002102435.1\_Ovir.te\_1.0/GCF\_002102435.1\_Ovir.te\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/007/570/765/GCA\_007570765.1\_GSC\_moose\_1.0/GCA\_007570765.1\_GSC\_moose\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/121/395/GCF\_003121395.1\_ASM312139v1/GCF\_003121395.1\_ASM312139v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/263/795/GCF\_002263795.1\_ARS-UCD1.2/GCF\_002263795.1\_ARS-UCD1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/754/665/GCF\_000754665.1\_Bison\_UMD1.0/GCF\_000754665.1\_Bison\_UMD1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/408/695/GCA\_006408695.1\_IMP/GCA\_006408695.1\_IMP\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/758/055/GCA\_009758055.1\_ASM975805v1/GCA\_009758055.1\_ASM975805v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/408/615/GCA\_006408615.1\_BWD/GCA\_006408615.1\_BWD\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/704/415/GCF\_001704415.1\_ARS1/GCF\_001704415.1\_ARS1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/742/125/GCF\_002742125.1\_Oar\_rambouillet\_v1.0/GCF\_002742125.1\_Oar\_rambouillet\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/027/775/GCA\_004027775.1\_BraVar\_v1\_BIUU/GCA\_004027775.1\_BraVar\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/208/655/GCF\_000208655.1\_Dasnov3.0/GCF\_000208655.1\_Dasnov3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/026/745/GCA\_004026745.1\_MyrTri\_v1\_BIUU/GCA\_004026745.1\_MyrTri\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/026/845/GCA\_004026845.1\_HetBruBak\_v1\_BIUU/GCA\_004026845.1\_HetBruBak\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/332/765/GCA\_014332765.1\_ASM1433276v1/GCA\_014332765.1\_ASM1433276v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/905/GCF\_000001905.1\_Loxafr3.0/GCF\_000001905.1\_Loxafr3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/985/GCF\_000313985.2\_ASM31398v2/GCF\_000313985.2\_ASM31398v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/298/275/GCF\_000298275.1\_OryAfe1.0/GCF\_000298275.1\_OryAfe1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/099/425/GCF\_002099425.1\_phaCin\_unsw\_v4.1/GCF\_002099425.1\_phaCin\_unsw\_v4.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/900/497/805/GCF\_900497805.2\_bare-nosed\_wombat\_genome\_assembly/GCF\_900497805.2\_bare-nosed\_wombat\_genome\_assembly\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/902/635/505/GCF\_902635505.1\_mSarHar1.11/GCF\_902635505.1\_mSarHar1.11\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/007/646/695/GCA\_007646695.1\_UniMelb\_Thylacine\_Refassem\_1/GCA\_007646695.1\_UniMelb\_Thylacine\_Refassem\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/004/035/GCA\_000004035.1\_Meug\_1.1/GCA\_000004035.1\_Meug\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/295/GCF\_000002295.2\_MonDom5/GCF\_000002295.2\_MonDom5\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/115/215/GCF\_004115215.1\_mOrnAna1.p.v1/GCF\_004115215.1\_mOrnAna1.p.v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/015/852/505/GCF\_015852505.1\_mTacAcu1.pri/GCF\_015852505.1\_mTacAcu1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/113/815/GCA\_003113815.1\_ASM311381v1/GCA\_003113815.1\_ASM311381v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/118/565/GCA\_003118565.1\_Ppicta\_assembly\_v1/GCA\_003118565.1\_Ppicta\_assembly\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/090/745/GCF\_000090745.1\_AnoCar2.0/GCF\_000090745.1\_AnoCar2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/245/905/GCA\_900245905.1\_ASM90024590v1/GCA\_900245905.1\_ASM90024590v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/090/745/GCF\_000090745.1\_AnoCar2.0/GCF\_000090745.1\_AnoCar2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/900/067/755/GCF\_900067755.1\_pvi1.1/GCF\_900067755.1\_pvi1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/798/865/GCA\_004798865.1\_ASM479886v1/GCA\_004798865.1\_ASM479886v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/012/654/085/GCA\_012654085.1\_UNIGE\_PanObs\_1.0/GCA\_012654085.1\_UNIGE\_PanObs\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/186/305/GCF\_000186305.1\_Python\_molurus\_bivittatus-5.0.2/GCF\_000186305.1\_Python\_molurus\_bivittatus-5.0.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/733/165/GCA\_009733165.1\_Nana\_v5/GCA\_009733165.1\_Nana\_v5\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/728/815/GCA\_001728815.2\_terp\_v2\_2/GCA\_001728815.2\_terp\_v2\_2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/597/395/GCF\_003597395.1\_ASM359739v1/GCF\_003597395.1\_ASM359739v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/241/765/GCF\_000241765.3\_Chrysemys\_picta\_bellii-3.0.3/GCF\_000241765.3\_Chrysemys\_picta\_bellii-3.0.3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/009/764/565/GCF\_009764565.2\_rDerCor1.pri.v3/GCF\_009764565.2\_rDerCor1.pri.v3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/161/935/GCA\_016161935.1\_ASM1616193v1/GCA\_016161935.1\_ASM1616193v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/007/922/165/GCA\_007922165.1\_Chelydra\_serpentina-1.0/GCA\_007922165.1\_Chelydra\_serpentina-1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/723/915/GCF\_001723915.1\_GavGan\_comp1/GCF\_001723915.1\_GavGan\_comp1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/455/745/GCF\_000455745.1\_ASM45574v1/GCF\_000455745.1\_ASM45574v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/281/125/GCF\_000281125.3\_ASM28112v4/GCF\_000281125.3\_ASM28112v4\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/723/895/GCF\_001723895.1\_CroPor\_comp1/GCF\_001723895.1\_CroPor\_comp1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/015/476/345/GCF\_015476345.1\_ZJU1.0/GCF\_015476345.1\_ZJU1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/130/075/GCA\_006130075.1\_GSC\_cangoose\_1.0/GCA\_006130075.1\_GSC\_cangoose\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/769/625/GCA\_009769625.1\_bCygOlo1.pri/GCA\_009769625.1\_bCygOlo1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/399/715/GCA\_013399715.1\_ASM1339971v1/GCA\_013399715.1\_ASM1339971v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/870/855/GCA\_001870855.1\_T\_cupido\_pinnatus\_GPC\_3440\_v1/GCA\_001870855.1\_T\_cupido\_pinnatus\_GPC\_3440\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/577/835/GCF\_001577835.2\_Coturnix\_japonica\_2.1/GCF\_001577835.2\_Coturnix\_japonica\_2.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/143/745/GCF\_004143745.1\_ASM414374v1/GCF\_004143745.1\_ASM414374v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/315/GCF\_000002315.6\_GRCg6a/GCF\_000002315.6\_GRCg6a\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/698/965/GCF\_000698965.1\_ASM69896v1/GCF\_000698965.1\_ASM69896v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/342/985/GCA\_003342985.1\_aptHaa1/GCA\_003342985.1\_aptHaa1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/398/345/GCA\_013398345.1\_ASM1339834v1/GCA\_013398345.1\_ASM1339834v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/396/415/GCA\_013396415.1\_ASM1339641v1/GCA\_013396415.1\_ASM1339641v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/398/225/GCA\_013398225.1\_ASM1339822v1/GCA\_013398225.1\_ASM1339822v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/396/855/GCA\_013396855.1\_ASM1339685v1/GCA\_013396855.1\_ASM1339685v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/957/555/GCF\_003957555.1\_bCalAnn1\_v1.p/GCF\_003957555.1\_bCalAnn1\_v1.p\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/819/775/GCA\_009819775.1\_bPhoRub2.pri/GCA\_009819775.1\_bPhoRub2.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/692/075/GCF\_000692075.1\_ASM69207v1/GCF\_000692075.1\_ASM69207v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/337/935/GCF\_000337935.1\_Cliv\_1.0/GCF\_000337935.1\_Cliv\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/372/525/GCA\_013372525.1\_Fatr\_1.0/GCA\_013372525.1\_Fatr\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/002/965/GCA\_002002965.1\_Ciconia\_boyciana\_ver1.0/GCA\_002002965.1\_Ciconia\_boyciana\_ver1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/708/925/GCF\_000708925.1\_ASM70892v1/GCF\_000708925.1\_ASM70892v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/077/935/GCA\_010077935.1\_BGI\_Sdem.V1/GCA\_010077935.1\_BGI\_Sdem.V1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/819/605/GCA\_009819605.1\_bSteHir1.pri/GCA\_009819605.1\_bSteHir1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/400/295/GCA\_013400295.1\_ASM1340029v1/GCA\_013400295.1\_ASM1340029v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/699/945/GCA\_000699945.1\_ASM69994v1/GCA\_000699945.1\_ASM69994v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/900/496/995/GCF\_900496995.1\_bAquChr1.2/GCF\_900496995.1\_bAquChr1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/902/150/015/GCF\_902150015.1\_genome\_assembly\_l500/GCF\_902150015.1\_genome\_assembly\_l500\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/303/855/GCA\_010303855.1\_BubBub1.0/GCA\_010303855.1\_BubBub1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/710/305/GCF\_000710305.1\_ASM71030v1/GCF\_000710305.1\_ASM71030v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/401/355/GCA\_013401355.1\_ASM1340135v1/GCA\_013401355.1\_ASM1340135v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/125/475/GCA\_011125475.1\_OU\_Melaur\_1/GCA\_011125475.1\_OU\_Melaur\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/399/055/GCA\_013399055.1\_ASM1339905v1/GCA\_013399055.1\_ASM1339905v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/690/535/GCF\_000690535.1\_ASM69053v1/GCF\_000690535.1\_ASM69053v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/337/955/GCF\_000337955.1\_F\_peregrinus\_v1.0/GCF\_000337955.1\_F\_peregrinus\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/700/915/GCA\_001700915.1\_Passer\_domesticus-1.0/GCA\_001700915.1\_Passer\_domesticus-1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/738/735/GCF\_000738735.2\_ASM73873v2/GCF\_000738735.2\_ASM73873v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/400/695/GCA\_000400695.1\_SMACv1.1/GCA\_000400695.1\_SMACv1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/700/915/GCA\_001700915.1\_Passer\_domesticus-1.0/GCA\_001700915.1\_Passer\_domesticus-1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/337/955/GCF\_000337955.1\_F\_peregrinus\_v1.0/GCF\_000337955.1\_F\_peregrinus\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/343/005/GCA\_003343005.1\_rheAme1/GCA\_003343005.1\_rheAme1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/905/171/775/GCF\_905171775.1\_aRanTem1.1/GCF\_905171775.1\_aRanTem1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/760/885/GCA\_009760885.1\_ASM976088v1/GCA\_009760885.1\_ASM976088v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/155/875/GCA\_014155875.1\_NBC\_Cnem\_1.0/GCA\_014155875.1\_NBC\_Cnem\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/999/395/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/985/GCF\_000002985.6\_WBcel235/GCF\_000002985.6\_WBcel235\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/082/055/GCA\_002082055.1\_nHd\_3.1/GCA\_002082055.1\_nHd\_3.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/024/985/GCA\_003024985.2\_Erow\_1.0/GCA\_003024985.2\_Erow\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/484/575/GCA\_000484575.1\_M\_martensii\_Version\_1/GCA\_000484575.1\_M\_martensii\_Version\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/235/015/GCA\_013235015.1\_Ave\_3.0/GCA\_013235015.1\_Ave\_3.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/085/665/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/239/455/GCA\_000239455.1\_Smar\_1.0/GCA\_000239455.1\_Smar\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/389/805/GCA\_013389805.1\_ASM1338980v1/GCA\_013389805.1\_ASM1338980v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/176/605/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/870/185/GCA\_009870185.1\_ASM987018v1/GCA\_009870185.1\_ASM987018v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/757/345/GCA\_009757345.1\_UGE\_Caugens\_1.0/GCA\_009757345.1\_UGE\_Caugens\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/456/935/GCA\_003456935.1\_Mhra\_1.0/GCA\_003456935.1\_Mhra\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/376/725/GCA\_000376725.2\_Lful\_2.0/GCA\_000376725.2\_Lful\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/507/165/GCA\_000507165.2\_Edan\_2.0/GCA\_000507165.2\_Edan\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/170/035/GCA\_011170035.1\_Tocci\_1.0/GCA\_011170035.1\_Tocci\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/858/955/GCA\_014858955.1\_NU\_Adom\_1.1/GCA\_014858955.1\_NU\_Adom\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/155/825/GCA\_902155825.1\_5S\_Tge\_b3v08/GCA\_902155825.1\_5S\_Tge\_b3v08\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/018/175/GCA\_003018175.1\_Bger\_1.1/GCA\_003018175.1\_Bger\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/891/405/GCF\_002891405.2\_Csec\_1.0/GCF\_002891405.2\_Csec\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/416/935/GCA\_010416935.1\_CU\_Pfus\_HIC/GCA\_010416935.1\_CU\_Pfus\_HIC\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/254/395/GCF\_003254395.2\_Amel\_HAv3.1/GCF\_003254395.2\_Amel\_HAv3.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/214/255/GCF\_000214255.1\_Bter\_1.0/GCF\_000214255.1\_Bter\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/651/465/GCF\_003651465.1\_ASM365146v1/GCF\_003651465.1\_ASM365146v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/217/595/GCF\_000217595.1\_Lhum\_UMD\_V04/GCF\_000217595.1\_Lhum\_UMD\_V04\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/005/508/785/GCF\_005508785.1\_pea\_aphid\_22Mar2018\_4r6ur/GCF\_005508785.1\_pea\_aphid\_22Mar2018\_4r6ur\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/009/605/GCF\_000009605.1\_ASM960v1/GCF\_000009605.1\_ASM960v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/005/845/GCF\_000005845.2\_ASM584v2/GCF\_000005845.2\_ASM584v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/326/945/GCA\_011326945.1\_ASM1132694v1/GCA\_011326945.1\_ASM1132694v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/786/065/GCA\_000786065.1\_Piez\_guildinii\_1/GCA\_000786065.1\_Piez\_guildinii\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/009/731/565/GCF\_009731565.1\_Dplex\_v4/GCF\_009731565.1\_Dplex\_v4\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/014/905/235/GCF\_014905235.1\_Bmori\_2016v1.0/GCF\_014905235.1\_Bmori\_2016v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/648/695/GCF\_000648695.1\_Otau\_2.0/GCF\_000648695.1\_Otau\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/568/925/GCA\_003568925.1\_Csep\_MD8\_v1/GCA\_003568925.1\_Csep\_MD8\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/215/GCF\_000001215.4\_Release\_6\_plus\_ISO1\_MT/GCF\_000001215.4\_Release\_6\_plus\_ISO1\_MT\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/371/365/GCF\_000371365.1\_Musca\_domestica-2.0.2/GCF\_000371365.1\_Musca\_domestica-2.0.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/015/586/225/GCF\_015586225.1\_ASM1558622v1/GCF\_015586225.1\_ASM1558622v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

}

async function marine\_genomes() {

unused\_variable = 5;

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/201/575/GCF\_002201575.1\_ASM220157v1/GCF\_002201575.1\_ASM220157v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/012/393/455/GCF\_012393455.1\_Tufts\_HGry\_1.1/GCF\_012393455.1\_Tufts\_HGry\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/023/865/GCA\_004023865.1\_MirAng\_v1\_BIUU/GCA\_004023865.1\_MirAng\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/321/225/GCF\_000321225.1\_Oros\_1.0/GCF\_000321225.1\_Oros\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/028/035/GCF\_004028035.1\_ASM402803v1/GCF\_004028035.1\_ASM402803v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/007/760/645/GCA\_007760645.1\_ASM776064v1/GCA\_007760645.1\_ASM776064v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/004/028/035/GCF\_004028035.1\_ASM402803v1/GCF\_004028035.1\_ASM402803v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/189/225/GCA\_002189225.1\_ASM218922v1/GCA\_002189225.1\_ASM218922v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/493/695/GCF\_000493695.1\_BalAcu1.0/GCF\_000493695.1\_BalAcu1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/837/175/GCF\_002837175.2\_ASM283717v2/GCF\_002837175.2\_ASM283717v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/329/385/GCA\_004329385.1\_megNov1/GCA\_004329385.1\_megNov1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/005/190/385/GCF\_005190385.1\_NGI\_Narwhal\_1/GCF\_005190385.1\_NGI\_Narwhal\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/331/955/GCF\_000331955.2\_Oorc\_1.1/GCF\_000331955.2\_Oorc\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/227/395/GCA\_003227395.1\_ASM322739v1/GCA\_003227395.1\_ASM322739v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/754/075/GCA\_011754075.1\_ASM1175407v1/GCA\_011754075.1\_ASM1175407v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/071/005/GCA\_003071005.2\_ASM307100v2/GCA\_003071005.2\_ASM307100v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/008/692/025/GCF\_008692025.1\_mPhoSin1.pri/GCF\_008692025.1\_mPhoSin1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/633/615/GCF\_000633615.1\_Guppy\_female\_1.0\_MT/GCF\_000633615.1\_Guppy\_female\_1.0\_MT\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/403/625/GCA\_013403625.1\_Bspl.v1.2018/GCA\_013403625.1\_Bspl.v1.2018\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/706/435/GCA\_014706435.1\_S\_gla\_Hod\_UTU\_1.0/GCA\_014706435.1\_S\_gla\_Hod\_UTU\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/011/004/845/GCF\_011004845.1\_fEsoLuc1.pri/GCF\_011004845.1\_fEsoLuc1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/242/695/GCF\_000242695.1\_LepOcu1/GCF\_000242695.1\_LepOcu1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/013/358/815/GCF\_013358815.1\_fEleEle1.pri/GCF\_013358815.1\_fEleEle1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/597/225/GCA\_003597225.1\_Ajp\_01/GCA\_003597225.1\_Ajp\_01\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/010/645/085/GCF\_010645085.1\_ASM1064508v1/GCF\_010645085.1\_ASM1064508v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/035/GCF\_000002035.6\_GRCz11/GCF\_000002035.6\_GRCz11\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/951/615/GCF\_000951615.1\_common\_carp\_genome/GCF\_000951615.1\_common\_carp\_genome\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/901/001/165/GCF\_901001165.1\_fSalTru1.1/GCF\_901001165.1\_fSalTru1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/819/105/GCA\_002819105.1\_ASM281910v1/GCA\_002819105.1\_ASM281910v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/776/465/GCF\_002776465.1\_AmpOce1.0/GCF\_002776465.1\_AmpOce1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/660/295/GCA\_009660295.1\_BF2\_Nord/GCA\_009660295.1\_BF2\_Nord\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/900/700/415/GCF\_900700415.1\_Ch\_v2.0.2/GCF\_900700415.1\_Ch\_v2.0.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/231/725/GCA\_003231725.1\_GBYP\_Tthy\_1.0/GCA\_003231725.1\_GBYP\_Tthy\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/499/035/GCA\_900499035.1\_SP\_G/GCA\_900499035.1\_SP\_G\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/086/565/GCA\_008086565.1\_UOO\_Thbifa\_1.2/GCA\_008086565.1\_UOO\_Thbifa\_1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/271/365/GCA\_016271365.1\_neoFor\_v3/GCA\_016271365.1\_neoFor\_v3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/225/785/GCF\_000225785.1\_LatCha1/GCF\_000225785.1\_LatCha1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/427/355/GCA\_003427355.1\_Storazame\_v1.0/GCA\_003427355.1\_Storazame\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/902/204/185/GCA\_902204185.2\_broker\_test2/GCA\_902204185.2\_broker\_test2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/642/345/GCF\_001642345.1\_ASM164234v2/GCF\_001642345.1\_ASM164234v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/238/235/GCA\_000238235.1\_LER\_WGS\_1/GCA\_000238235.1\_LER\_WGS\_1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/010/993/605/GCF\_010993605.1\_kPetMar1.pri/GCF\_010993605.1\_kPetMar1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/621/495/GCA\_014621495.2\_ETRf\_v1/GCA\_014621495.2\_ETRf\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/186/335/GCA\_900186335.2\_Eburgeri\_3.2/GCA\_900186335.2\_Eburgeri\_3.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/224/145/GCF\_000224145.3\_KH/GCF\_000224145.3\_KH\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/013/122/585/GCF\_013122585.1\_ASM1312258v2/GCF\_013122585.1\_ASM1312258v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/749/815/GCA\_001749815.1\_Salp\_genome\_1.0/GCA\_001749815.1\_Salp\_genome\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/367/955/GCA\_004367955.1\_ASM436795v1/GCA\_004367955.1\_ASM436795v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/663/935/GCA\_001663935.1\_Asyluc0.1/GCA\_001663935.1\_Asyluc0.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/003/815/GCF\_000003815.2\_Bfl\_VNyyK/GCF\_000003815.2\_Bfl\_VNyyK\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/088/365/GCA\_900088365.1\_Bl71nemr/GCA\_900088365.1\_Bl71nemr\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/003/605/GCF\_000003605.2\_Skow\_1.1/GCF\_000003605.2\_Skow\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/011/630/105/GCF\_011630105.1\_ASM1163010v1/GCF\_011630105.1\_ASM1163010v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/902/459/465/GCF\_902459465.1\_eAstRub1.3/GCF\_902459465.1\_eAstRub1.3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/015/706/575/GCF\_015706575.1\_ASM1570657v1/GCF\_015706575.1\_ASM1570657v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/067/615/GCA\_900067615.1\_Ophionereis\_fasciata\_genome\_assembly\_1.0/GCA\_900067615.1\_Ophionereis\_fasciata\_genome\_assembly\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/188/425/GCA\_001188425.1\_Etri\_1.0/GCA\_001188425.1\_Etri\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/235/GCF\_000002235.5\_Spur\_5.0/GCF\_000002235.5\_Spur\_5.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/015/342/785/GCA\_015342785.1\_UCSD\_Lpic\_2.0/GCA\_015342785.1\_UCSD\_Lpic\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/067/635/GCA\_900067635.1\_Australostichopus\_mollis\_genome\_assembly\_1.0/GCA\_900067635.1\_Australostichopus\_mollis\_genome\_assembly\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/015/985/GCA\_010015985.1\_ASM1001598v1/GCA\_010015985.1\_ASM1001598v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/760/885/GCA\_009760885.1\_ASM976088v1/GCA\_009760885.1\_ASM976088v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/457/365/GCF\_000457365.1\_ASM45736v1/GCF\_000457365.1\_ASM45736v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/800/805/GCA\_011800805.1\_ASM1180080v1/GCA\_011800805.1\_ASM1180080v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/999/395/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/485/595/GCF\_000485595.1\_Priapulus\_caudatus-5.0.1/GCF\_000485595.1\_Priapulus\_caudatus-5.0.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/985/GCF\_000002985.6\_WBcel235/GCF\_000002985.6\_WBcel235\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/082/055/GCA\_002082055.1\_nHd\_3.1/GCA\_002082055.1\_nHd\_3.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/024/985/GCA\_003024985.2\_Erow\_1.0/GCA\_003024985.2\_Erow\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/085/665/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/210/375/GCA\_004210375.1\_ASM421037v1/GCA\_004210375.1\_ASM421037v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/239/455/GCA\_000239455.1\_Smar\_1.0/GCA\_000239455.1\_Smar\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/376/725/GCA\_000376725.2\_Lful\_2.0/GCA\_000376725.2\_Lful\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/176/605/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/587/735/GCA\_001587735.2\_Phaw\_5.0/GCA\_001587735.2\_Phaw\_5.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/283/005/GCA\_013283005.1\_ASM1328300v1/GCA\_013283005.1\_ASM1328300v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/092/285/GCA\_900092285.2\_PA42\_4.1/GCA\_900092285.2\_PA42\_4.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/981/345/GCA\_000981345.1\_tcf\_1.0/GCA\_000981345.1\_tcf\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/095/GCF\_000004095.1\_Hydra\_RP\_1.0/GCF\_000004095.1\_Hydra\_RP\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/903/981/925/GCA\_903981925.1\_v081/GCA\_903981925.1\_v081\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/pearl/download/pfu\_genome2.0.fasta.zip");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://marinegenomics.oist.jp/aurelia\_aurita/download/AUR21\_r04\_250316.fa.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/lingula/download/lingula\_genome\_v1.0\_unmasked.fa.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/339/895/GCA\_013339895.1\_Emu\_genome\_v1/GCA\_013339895.1\_Emu\_genome\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/090/795/GCF\_000090795.1\_v1.0/GCF\_000090795.1\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/275/595/GCA\_900275595.1\_Aplysina23\_polished\_assembly/GCA\_900275595.1\_Aplysina23\_polished\_assembly\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/nge\_v2/download/nge\_genome\_v2.0.fa.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/634/525/GCA\_014634525.1\_Asel\_1.0/GCA\_014634525.1\_Asel\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/morbakka\_virulenta/download/MOR05\_r06\_genome.fa.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/660/155/GCA\_900660155.1\_ASM90066015v1/GCA\_900660155.1\_ASM90066015v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/239/995/GCA\_900239995.1\_ASM90023999v1/GCA\_900239995.1\_ASM90023999v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/150/275/GCF\_000150275.1\_v1.0/GCF\_000150275.1\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/865/GCF\_000002865.3\_V1.0/GCF\_000002865.3\_V1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/165/875/GCA\_016165875.1\_ASM1616587v1/GCA\_016165875.1\_ASM1616587v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/957/725/GCA\_003957725.1\_ASM395772v1/GCA\_003957725.1\_ASM395772v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/765/925/GCA\_004765925.1\_PBRC\_Esco\_1.0/GCA\_004765925.1\_PBRC\_Esco\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/352/455/GCA\_006352455.1\_Btisc\_TiscarSM28/GCA\_006352455.1\_Btisc\_TiscarSM28\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/279/815/GCA\_010279815.1\_ASM1027981v1/GCA\_010279815.1\_ASM1027981v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/760/855/GCA\_009760855.2\_Cmucedo\_v1.1/GCA\_009760855.2\_Cmucedo\_v1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/633/005/GCA\_002633005.1\_ASM263300v1/GCA\_002633005.1\_ASM263300v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/382/745/GCA\_004382745.1\_QAU\_Airr\_1.1/GCA\_004382745.1\_QAU\_Airr\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/600/895/GCA\_002600895.1\_ASM260089v1/GCA\_002600895.1\_ASM260089v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

}

async function freshwater\_genomes() {

unused\_variable = 5;

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/900/700/415/GCF\_900700415.1\_Ch\_v2.0.2/GCF\_900700415.1\_Ch\_v2.0.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/499/035/GCA\_900499035.1\_SP\_G/GCA\_900499035.1\_SP\_G\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/086/565/GCA\_008086565.1\_UOO\_Thbifa\_1.2/GCA\_008086565.1\_UOO\_Thbifa\_1.2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/002/776/465/GCF\_002776465.1\_AmpOce1.0/GCF\_002776465.1\_AmpOce1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/660/295/GCA\_009660295.1\_BF2\_Nord/GCA\_009660295.1\_BF2\_Nord\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/231/725/GCA\_003231725.1\_GBYP\_Tthy\_1.0/GCA\_003231725.1\_GBYP\_Tthy\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/819/105/GCA\_002819105.1\_ASM281910v1/GCA\_002819105.1\_ASM281910v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/455/525/GCA\_001455525.1\_AmbMex13\_14.1.0/GCA\_001455525.1\_AmbMex13\_14.1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/663/975/GCF\_001663975.1\_Xenopus\_laevis\_v2/GCF\_001663975.1\_Xenopus\_laevis\_v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/802/015/GCA\_009802015.1\_UCB\_Rtem\_1.0/GCA\_009802015.1\_UCB\_Rtem\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/901/765/095/GCF\_901765095.1\_aMicUni1.1/GCF\_901765095.1\_aMicUni1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/442/215/GCF\_000442215.1\_Lipotes\_vexillifer\_v1/GCF\_000442215.1\_Lipotes\_vexillifer\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/754/075/GCA\_011754075.1\_ASM1175407v1/GCA\_011754075.1\_ASM1175407v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/035/GCF\_000002035.6\_GRCz11/GCF\_000002035.6\_GRCz11\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/015/445/GCA\_010015445.1\_GENO\_Pfluv\_1.0/GCA\_010015445.1\_GENO\_Pfluv\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/233/375/GCF\_000233375.1\_ICSASG\_v2/GCF\_000233375.1\_ICSASG\_v2\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/901/001/165/GCF\_901001165.1\_fSalTru1.1/GCF\_901001165.1\_fSalTru1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/951/615/GCF\_000951615.1\_common\_carp\_genome/GCF\_000951615.1\_common\_carp\_genome\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/003/368/295/GCF\_003368295.1\_ASM336829v1/GCF\_003368295.1\_ASM336829v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/869/865/GCA\_009869865.1\_ASM986986v1/GCA\_009869865.1\_ASM986986v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/242/695/GCF\_000242695.1\_LepOcu1/GCF\_000242695.1\_LepOcu1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/011/004/845/GCF\_011004845.1\_fEsoLuc1.pri/GCF\_011004845.1\_fEsoLuc1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/706/435/GCA\_014706435.1\_S\_gla\_Hod\_UTU\_1.0/GCA\_014706435.1\_S\_gla\_Hod\_UTU\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/010/645/085/GCF\_010645085.1\_ASM1064508v1/GCF\_010645085.1\_ASM1064508v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/013/358/815/GCF\_013358815.1\_fEleEle1.pri/GCF\_013358815.1\_fEleEle1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/606/085/GCA\_001606085.1\_ASM160608v1/GCA\_001606085.1\_ASM160608v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/363/515/GCA\_004363515.1\_IniGeo\_v1\_BIUU/GCA\_004363515.1\_IniGeo\_v1\_BIUU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/597/225/GCA\_003597225.1\_Ajp\_01/GCA\_003597225.1\_Ajp\_01\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/015/108/615/GCA\_015108615.1\_JWE2.scf/GCA\_015108615.1\_JWE2.scf\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/633/615/GCF\_000633615.1\_Guppy\_female\_1.0\_MT/GCF\_000633615.1\_Guppy\_female\_1.0\_MT\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/271/365/GCA\_016271365.1\_neoFor\_v3/GCA\_016271365.1\_neoFor\_v3\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/010/993/605/GCF\_010993605.1\_kPetMar1.pri/GCF\_010993605.1\_kPetMar1.pri\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/015/708/825/GCA\_015708825.1\_ASM1570882v1/GCA\_015708825.1\_ASM1570882v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/466/285/GCA\_000466285.1\_LetJap1.0/GCA\_000466285.1\_LetJap1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/195/GCF\_000004195.4\_UCB\_Xtro\_10.0/GCF\_000004195.4\_UCB\_Xtro\_10.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/901/001/135/GCF\_901001135.1\_aRhiBiv1.1/GCF\_901001135.1\_aRhiBiv1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/014/858/855/GCA\_014858855.1\_ASM1485885v1/GCA\_014858855.1\_ASM1485885v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/225/785/GCF\_000225785.1\_LatCha1/GCF\_000225785.1\_LatCha1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/999/395/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP/GCA\_003999395.1\_Eisenia\_fetida\_CLC\_PE\_MP\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/760/885/GCA\_009760885.1\_ASM976088v1/GCA\_009760885.1\_ASM976088v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/985/GCF\_000002985.6\_WBcel235/GCF\_000002985.6\_WBcel235\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/082/055/GCA\_002082055.1\_nHd\_3.1/GCA\_002082055.1\_nHd\_3.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/024/985/GCA\_003024985.2\_Erow\_1.0/GCA\_003024985.2\_Erow\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/085/665/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1/GCA\_002085665.2\_Dfa\_Genome\_UMICH\_USM\_1.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/239/455/GCA\_000239455.1\_Smar\_1.0/GCA\_000239455.1\_Smar\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/176/605/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0/GCA\_009176605.1\_CNRS\_Arma\_nasa\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/376/725/GCA\_000376725.2\_Lful\_2.0/GCA\_000376725.2\_Lful\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/507/165/GCA\_000507165.2\_Edan\_2.0/GCA\_000507165.2\_Edan\_2.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/006/352/455/GCA\_006352455.1\_Btisc\_TiscarSM28/GCA\_006352455.1\_Btisc\_TiscarSM28\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/922/825/GCA\_002922825.1\_Bc\_v0.6/GCA\_002922825.1\_Bc\_v0.6\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/802/215/GCA\_016802215.1\_ASM1680221v1/GCA\_016802215.1\_ASM1680221v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/938/525/GCA\_001938525.1\_Djap\_assembly\_v1/GCA\_001938525.1\_Djap\_assembly\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/938/485/GCA\_001938485.1\_gtig.1/GCA\_001938485.1\_gtig.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/600/895/GCA\_002600895.1\_ASM260089v1/GCA\_002600895.1\_ASM260089v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/072/015/GCA\_002072015.1\_ASM207201v1/GCA\_002072015.1\_ASM207201v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/457/365/GCF\_000457365.1\_ASM45736v1/GCF\_000457365.1\_ASM45736v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/036/025/GCA\_900036025.1\_v1.0/GCA\_900036025.1\_v1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/038/615/GCA\_011038615.1\_BGI\_Ldum\_1.0/GCA\_011038615.1\_BGI\_Ldum\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/011/800/805/GCA\_011800805.1\_ASM1180080v1/GCA\_011800805.1\_ASM1180080v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/981/345/GCA\_000981345.1\_tcf\_1.0/GCA\_000981345.1\_tcf\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/900/092/285/GCA\_900092285.2\_PA42\_4.1/GCA\_900092285.2\_PA42\_4.1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/004/095/GCF\_000004095.1\_Hydra\_RP\_1.0/GCF\_000004095.1\_Hydra\_RP\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/013/339/895/GCA\_013339895.1\_Emu\_genome\_v1/GCA\_013339895.1\_Emu\_genome\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

evolutionary\_genomes.push(load);

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

evolutionary\_genomes.push(eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1]);

}

}

async function additional\_genomes() {

unused\_variable = 5;

if (unused\_variable == 5) {

genomobtained = await fetch("https://treegenesdb.org/FTP/Genomes/Gibi/v1.0/annotation/Gibi.1\_0.cds.fa.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://treegenesdb.org/FTP/Genomes/Gnmo/v1.0/genome/Gnmo.1\_0.fa.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://genomevolution.org/coge/api/v1/genomes/36052/sequence");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://genomevolution.org/coge/api/v1/genomes/36051/sequence");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/umibudo/download/clen\_genome\_scaffold-v1.1.fa.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/chlorellaA99/download/Ch\_genome\_seq\_final.mod01.fasta.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/chlorellaA99/download/Ch\_genome\_seq\_final.mod01.fasta.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/chlorellaA99/download/Ch\_genome\_seq\_final.mod01.fasta.gz");

load = await genomobtained.blob();

plant\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR229/ERR229910/ERR229910.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR257/ERR257718/ERR257718.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("http://cdna.eva.mpg.de/denisova/raw\_reads/B1130\_SR.txt.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR995/ERR995357/ERR995357.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR502/002/ERR5024912/ERR5024912\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR503/007/ERR5032057/ERR5032057\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR203/002/SRR2034332/SRR2034332\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR226/003/ERR2260503/ERR2260503.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR226/005/ERR2260495/ERR2260495.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR226/006/ERR2260506/ERR2260506.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR257/SRR257736/SRR257736.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR267/004/ERR2678614/ERR2678614\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR555/002/SRR5556842/SRR5556842\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR169/005/SRR1693815/SRR1693815\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR246/009/SRR2460709/SRR2460709.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR959/SRR959261/SRR959261\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR317/002/SRR3178392/SRR3178392.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR114/038/SRR11428438/SRR11428438\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR843/008/SRR8437788/SRR8437788\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR629/001/SRR6299281/SRR6299281.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR126/048/SRR12612948/SRR12612948\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR123/015/SRR12354115/SRR12354115.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR317/007/SRR3178397/SRR3178397.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR317/007/SRR3178397/SRR3178397.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR317/004/SRR3178404/SRR3178404.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR317/004/SRR3178404/SRR3178404.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/005/SRR3180905/SRR3180905.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/008/SRR3180908/SRR3180908.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/005/SRR3180905/SRR3180905.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/004/SRR3180914/SRR3180914.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/005/SRR3180975/SRR3180975.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/008/SRR3180998/SRR3180998.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/004/SRR3181004/SRR3181004.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR318/001/SRR3181051/SRR3181051.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR689/001/SRR6893011/SRR6893011\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR689/001/SRR6893011/SRR6893011\_2.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR347/000/ERR3473230/ERR3473230.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR347/003/ERR3473723/ERR3473723.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR362/001/ERR3623511/ERR3623511.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR362/006/ERR3624106/ERR3624106.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR362/007/ERR3624107/ERR3624107.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR130/003/SRR1303453/SRR1303453\_1.fastq.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://marinegenomics.oist.jp/amphidinium/download/83\_amphidinium\_scaffold.fa.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/symbd/download/102\_symbd\_genome\_scaffold.fa.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/ito\_mozuku\_v1/download/ito\_mozuku\_onna1\_genome\_with\_bacteria.fa.zip");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/symb/download/symC\_aug\_40.aa.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/symb/download/symA3\_37.fasta.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/symb/download/symA3\_37.fasta.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

genomobtained = await fetch("https://marinegenomics.oist.jp/symb/download/symA3\_37.fasta.gz");

load = await genomobtained.blob();

eukaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/005/845/GCF\_000005845.2\_ASM584v2/GCF\_000005845.2\_ASM584v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/745/415/GCA\_002745415.1\_ASM274541v1/GCA\_002745415.1\_ASM274541v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/852/145/GCF\_000852145.1\_ViralProj14792/GCF\_000852145.1\_ViralProj14792\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/826/345/GCA\_000826345.1\_Acanthamoeba\_polyphaga/GCA\_000826345.1\_Acanthamoeba\_polyphaga\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/365/GCA\_002966365.1\_ASM296636v1/GCA\_002966365.1\_ASM296636v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/826/345/GCA\_000826345.1\_Acanthamoeba\_polyphaga/GCA\_000826345.1\_Acanthamoeba\_polyphaga\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/826/345/GCA\_000826345.1\_Acanthamoeba\_polyphaga/GCA\_000826345.1\_Acanthamoeba\_polyphaga\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/151/645/GCA\_004151645.1\_ASM415164v1/GCA\_004151645.1\_ASM415164v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/837/025/GCA\_000837025.1\_ViralProj14062/GCA\_000837025.1\_ViralProj14062\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/005/845/GCF\_000005845.2\_ASM584v2/GCF\_000005845.2\_ASM584v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/857/505/GCA\_000857505.1\_ViralProj15239/GCA\_000857505.1\_ViralProj15239\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/855/GCF\_000313855.2\_ASM31385v2/GCF\_000313855.2\_ASM31385v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/675/295/GCA\_001675295.1\_FPOA1.0/GCA\_001675295.1\_FPOA1.0\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/001/722/785/GCA\_001722785.1\_ViralProj342443/GCA\_001722785.1\_ViralProj342443\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/847/405/GCA\_000847405.1\_ViralProj14643/GCA\_000847405.1\_ViralProj14643\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/005/845/GCF\_000005845.2\_ASM584v2/GCF\_000005845.2\_ASM584v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/010/120/155/GCA\_010120155.1\_ASM1012015v1/GCA\_010120155.1\_ASM1012015v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/893/915/GCA\_000893915.1\_ViralProj74349/GCA\_000893915.1\_ViralProj74349\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/911/955/GCA\_000911955.1\_ViralProj215788/GCA\_000911955.1\_ViralProj215788\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/916/835/GCA\_000916835.1\_ViralProj237323/GCA\_000916835.1\_ViralProj237323\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/425/GCA\_002966425.1\_ASM296642v1/GCA\_002966425.1\_ASM296642v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/233/895/GCA\_003233895.1\_ASM323389v1/GCA\_003233895.1\_ASM323389v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/475/GCA\_002966475.2\_ASM296647v2/GCA\_002966475.2\_ASM296647v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/485/GCA\_002966485.2\_ASM296648v2/GCA\_002966485.2\_ASM296648v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/918/435/GCF\_000918435.1\_ViralProj240235/GCF\_000918435.1\_ViralProj240235\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/880/395/GCF\_000880395.1\_ViralProj30929/GCF\_000880395.1\_ViralProj30929\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/313/135/GCF\_000313135.1\_Acastellanii.strNEFF\_v1/GCF\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/330/645/GCA\_008330645.1\_CrBVI/GCA\_008330645.1\_CrBVI\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/889/395/GCF\_000889395.1\_ViralProj59783/GCF\_000889395.1\_ViralProj59783\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/954/985/GCA\_002954985.1\_ASM295498v1/GCA\_002954985.1\_ASM295498v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/330/645/GCA\_008330645.1\_CrBVI/GCA\_008330645.1\_CrBVI\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/889/395/GCF\_000889395.1\_ViralProj59783/GCF\_000889395.1\_ViralProj59783\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/954/985/GCA\_002954985.1\_ASM295498v1/GCA\_002954985.1\_ASM295498v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/330/645/GCA\_008330645.1\_CrBVI/GCA\_008330645.1\_CrBVI\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/954/985/GCA\_002954985.1\_ASM295498v1/GCA\_002954985.1\_ASM295498v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomeobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/890/715/GCA\_000890715.1\_ViralProj64497/GCA\_000890715.1\_ViralProj64497\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/335/GCA\_002966335.1\_ASM296633v1/GCA\_002966335.1\_ASM296633v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/313/135/GCA\_000313135.1\_Acastellanii.strNEFF\_v1/GCA\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/915/035/GCF\_000915035.1\_ViralProj230580/GCF\_000915035.1\_ViralProj230580\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/966/425/GCA\_002966425.1\_ASM296642v1/GCA\_002966425.1\_ASM296642v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/313/135/GCA\_000313135.1\_Acastellanii.strNEFF\_v1/GCA\_000313135.1\_Acastellanii.strNEFF\_v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/003/006/235/GCA\_003006235.2\_ASM300623v2/GCA\_003006235.2\_ASM300623v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/representative/GCA\_900617955.1\_HIFI\_SE400\_contigs.fasta/GCA\_900617955.1\_HIFI\_SE400\_contigs.fasta\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_Escherichia\_coli\_Syn61/all\_assembly\_versions/GCA\_019397285.1\_ASM1939728v1/GCA\_019397285.1\_ASM1939728v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_Mycoplasma\_genitalium\_JCVI-1.0/all\_assembly\_versions/GCA\_000019105.1\_ASM1910v1/GCA\_000019105.1\_ASM1910v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_Mycoplasma\_mycoides\_JCVI-syn1.0/all\_assembly\_versions/GCA\_000091445.1\_ASM9144v1/GCA\_000091445.1\_ASM9144v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_Mycoplasma\_mycoides\_JCVI-syn1.0/all\_assembly\_versions/GCA\_000091445.1\_ASM9144v1/GCA\_000091445.1\_ASM9144v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_bacterium\_JCVI-Syn3.0/all\_assembly\_versions/GCA\_001708325.1\_ASM170832v1/GCA\_001708325.1\_ASM170832v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_bacterium\_JCVI-Syn3A/all\_assembly\_versions/GCA\_001708325.2\_ASM170832v2/GCA\_001708325.2\_ASM170832v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_bacterium\_JCVI-Syn3B/all\_assembly\_versions/GCA\_019289195.1\_ASM1928919v1/GCA\_019289195.1\_ASM1928919v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/synthetic\_bacterium\_JCVI-syn3B-P026/all\_assembly\_versions/GCA\_018831505.1\_ASM1883150v1/GCA\_018831505.1\_ASM1883150v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/test\_organism/latest\_assembly\_versions/GCA\_018127685.2\_TEST\_org\_raw\_chrv2\_IGNORE/GCA\_018127685.2\_TEST\_org\_raw\_chrv2\_IGNORE\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified/all\_assembly\_versions/GCA\_902726115.1\_GD\_UNKNOWN\_CIR\_63\_59/GCA\_902726115.1\_GD\_UNKNOWN\_CIR\_63\_59\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified/all\_assembly\_versions/GCA\_902726205.1\_N5\_UNKNOWN\_CIR\_33\_1450/GCA\_902726205.1\_N5\_UNKNOWN\_CIR\_33\_1450\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified/all\_assembly\_versions/GCA\_902726475.1\_RIF\_UNKNOWN\_CIR\_35\_23/GCA\_902726475.1\_RIF\_UNKNOWN\_CIR\_35\_23\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified/all\_assembly\_versions/GCA\_902731675.1\_STOOL\_UNKNOWN\_CIR\_37\_17/GCA\_902731675.1\_STOOL\_UNKNOWN\_CIR\_37\_17\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified\_plasmid/all\_assembly\_versions/GCA\_902726495.1\_S40\_PLASMID\_36\_44/GCA\_902726495.1\_S40\_PLASMID\_36\_44\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified\_plasmid/all\_assembly\_versions/GCA\_902731075.1\_R2\_PLASMID\_CIR\_59\_76/GCA\_902731075.1\_R2\_PLASMID\_CIR\_59\_76\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified\_plasmid/all\_assembly\_versions/GCA\_902731205.1\_S9\_PLASMID\_CIR\_39\_1/GCA\_902731205.1\_S9\_PLASMID\_CIR\_39\_1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified\_plasmid/all\_assembly\_versions/GCA\_902731275.1\_S25\_PLASMID\_36\_160/GCA\_902731275.1\_S25\_PLASMID\_36\_160\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified\_plasmid/all\_assembly\_versions/GCA\_902731715.1\_BJP\_PLASMID\_CIR\_60\_13/");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-331-C09/all\_assembly\_versions/GCA\_003282675.1\_ASM328267v1/GCA\_003282675.1\_ASM328267v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-331-K15/all\_assembly\_versions/GCA\_003282505.1\_ASM328250v1/GCA\_003282505.1\_ASM328250v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-J21/all\_assembly\_versions/GCA\_003281945.1\_ASM328194v1/GCA\_003281945.1\_ASM328194v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-M17/all\_assembly\_versions/GCA\_003279275.1\_ASM327927v1/GCA\_003279275.1\_ASM327927v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-O18/all\_assembly\_versions/GCA\_003281895.1\_ASM328189v1/GCA\_003281895.1\_ASM328189v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-P21/all\_assembly\_versions/GCA\_003281875.1\_ASM328187v1/GCA\_003281875.1\_ASM328187v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/Saccharomyces\_cerevisiae\_synthetic\_construct/all\_assembly\_versions/GCA\_002072045.1\_ASM207204v1/GCA\_002072045.1\_ASM207204v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230595.1\_oilsands\_bin\_005/GCA\_900230595.1\_oilsands\_bin\_005\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230605.1\_oilsands\_bin\_010/GCA\_900230605.1\_oilsands\_bin\_010\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230615.1\_oilsands\_bin\_006/GCA\_900230615.1\_oilsands\_bin\_006\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230625.1\_oilsands\_bin\_019/GCA\_900230625.1\_oilsands\_bin\_019\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230635.1\_oilsands\_bin\_020/GCA\_900230635.1\_oilsands\_bin\_020\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230645.1\_oilsands\_bin\_023/GCA\_900230645.1\_oilsands\_bin\_023\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230655.1\_oilsands\_bin\_024/GCA\_900230655.1\_oilsands\_bin\_024\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230665.1\_oilsands\_bin\_039/GCA\_900230665.1\_oilsands\_bin\_039\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230675.1\_oilsands\_bin\_034/GCA\_900230675.1\_oilsands\_bin\_034\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/all\_assembly\_versions/GCA\_900230685.1\_oilsands\_bin\_049/GCA\_900230685.1\_oilsands\_bin\_049\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/eukaryotic\_synthetic\_construct/all\_assembly\_versions/GCA\_004000535.1\_ASM400053v1/GCA\_004000535.1\_ASM400053v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/all\_assembly\_versions/GCA\_002754765.1\_ASM275476v1/GCA\_002754765.1\_ASM275476v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/all\_assembly\_versions/GCA\_900214915.1\_SRR5275893/GCA\_900214915.1\_SRR5275893\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/all\_assembly\_versions/GCA\_900617955.1\_HIFI\_SE400\_contigs.fasta/GCA\_900617955.1\_HIFI\_SE400\_contigs.fasta\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/Saccharomyces\_cerevisiae\_synthetic\_construct/all\_assembly\_versions/GCA\_002072045.1\_ASM207204v1/GCA\_002072045.1\_ASM207204v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

}

if (unused\_variable == 5) {

genomeobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/012/932/795/GCF\_012932795.1\_ASM1293279v1/GCF\_012932795.1\_ASM1293279v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/activated\_carbon\_metagenome/latest\_assembly\_versions/GCA\_001468985.1\_PRJNA301005\_A2\_filter\_metagenome\_assembly/GCA\_001468985.1\_PRJNA301005\_A2\_filter\_metagenome\_assembly\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/activated\_sludge\_metagenome/latest\_assembly\_versions/GCA\_000803365.1\_AK-R06\_Meta\_assembly/GCA\_000803365.1\_AK-R06\_Meta\_assembly\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/rock\_metagenome/latest\_assembly\_versions/GCA\_010102235.1\_ASM1010223v1/GCA\_010102235.1\_ASM1010223v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/soil\_metagenome/latest\_assembly\_versions/GCA\_001482255.1\_Rock\_Bay\_complete\_shotgun\_assembly/GCA\_001482255.1\_Rock\_Bay\_complete\_shotgun\_assembly\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/wetland\_metagenome/latest\_assembly\_versions/GCA\_013548415.1\_ASM1354841v1/GCA\_013548415.1\_ASM1354841v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/enrichment\_culture/latest\_assembly\_versions/GCA\_900230645.1\_oilsands\_bin\_023/GCA\_900230645.1\_oilsands\_bin\_023\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/unidentified/latest\_assembly\_versions/GCA\_902726475.1\_RIF\_UNKNOWN\_CIR\_35\_23/GCA\_902726475.1\_RIF\_UNKNOWN\_CIR\_35\_23\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/latest\_assembly\_versions/GCA\_900214915.1\_SRR5275893/GCA\_900214915.1\_SRR5275893\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-331-C09/latest\_assembly\_versions/GCA\_003282675.1\_ASM328267v1/GCA\_003282675.1\_ASM328267v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-331-K15/representative/GCA\_003282505.1\_ASM328250v1/GCA\_003282505.1\_ASM328250v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-J21/latest\_assembly\_versions/GCA\_003281945.1\_ASM328194v1/GCA\_003281945.1\_ASM328194v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-M17/latest\_assembly\_versions/GCA\_003279275.1\_ASM327927v1/GCA\_003279275.1\_ASM327927v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-O18/representative/GCA\_003281895.1\_ASM328189v1/GCA\_003281895.1\_ASM328189v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/marine\_microorganism\_AG-341-P21/representative/GCA\_003281875.1\_ASM328187v1/GCA\_003281875.1\_ASM328187v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_004172875.1\_ASM417287v1/GCA\_004172875.1\_ASM417287v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009908905.1\_ASM990890v1/GCA\_009908905.1\_ASM990890v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009993545.1\_ASM999354v1/GCA\_009993545.1\_ASM999354v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009995505.1\_ASM999550v1/GCA\_009995505.1\_ASM999550v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_011088145.1\_ASM1108814v1/GCA\_011088145.1\_ASM1108814v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013203655.1\_ASM1320365v1/GCA\_013203655.1\_ASM1320365v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013205255.1\_ASM1320525v1/GCA\_013205255.1\_ASM1320525v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/lake\_water\_metagenome/latest\_assembly\_versions/GCA\_008633385.1\_ASM863338v1/GCA\_008633385.1\_ASM863338v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/decomposition\_metagenome/latest\_assembly\_versions/GCA\_902144195.1\_Cellvibrio\_fibrivorans\_DSM\_23887/GCA\_902144195.1\_Cellvibrio\_fibrivorans\_DSM\_23887\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/cold\_seep\_metagenome/latest\_assembly\_versions/GCA\_006844845.1\_ASM684484v1/GCA\_006844845.1\_ASM684484v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/compost\_metagenome/latest\_assembly\_versions/GCA\_000205965.1\_ASM20596v1/GCA\_000205965.1\_ASM20596v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/beach\_sand\_metagenome/latest\_assembly\_versions/GCA\_900216925.1\_SRR1570765/GCA\_900216925.1\_SRR1570765\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/biofilm\_metagenome/latest\_assembly\_versions/GCA\_002919965.1\_MiSeq/GCA\_002919965.1\_MiSeq\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/algae\_metagenome/all\_assembly\_versions/GCA\_013687095.1\_ASM1368709v1/GCA\_013687095.1\_ASM1368709v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/anaerobic\_digester\_metagenome/all\_assembly\_versions/GCA\_001509935.1\_R1meso\_AssemblyC/GCA\_001509935.1\_R1meso\_AssemblyC\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/bioreactor\_metagenome/all\_assembly\_versions/GCA\_000401835.1\_bioreactorBK5\_day119/GCA\_000401835.1\_bioreactorBK5\_day119\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/bioreactor\_sludge\_metagenome/latest\_assembly\_versions/GCA\_001945615.1\_ThermoCellSludge1.0/GCA\_001945615.1\_ThermoCellSludge1.0\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/whale\_fall\_metagenome/latest\_assembly\_versions/GCA\_000206205.1\_ASM20620v1/GCA\_000206205.1\_ASM20620v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/saltern\_metagenome/latest\_assembly\_versions/GCA\_900492215.1\_S56/GCA\_900492215.1\_S56\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/glacier\_metagenome/latest\_assembly\_versions/GCA\_001029185.1\_ASM102918v1/GCA\_001029185.1\_ASM102918v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/subsurface\_metagenome/latest\_assembly\_versions/GCA\_013282415.1\_ASM1328241v1/GCA\_013282415.1\_ASM1328241v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/stromatolite\_metagenome/latest\_assembly\_versions/GCA\_000207665.1\_ASM20766v1/GCA\_000207665.1\_ASM20766v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/terrestrial\_metagenome/latest\_assembly\_versions/GCA\_001507825.1\_ASM150782v1/GCA\_001507825.1\_ASM150782v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/rock\_porewater\_metagenome/latest\_assembly\_versions/GCA\_000961915.2\_ASM96191v2/GCA\_000961915.2\_ASM96191v2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/indoor\_metagenome/latest\_assembly\_versions/GCA\_900682105.1\_Fplus10/GCA\_900682105.1\_Fplus10\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/groundwater\_metagenome/latest\_assembly\_versions/GCA\_900206355.1\_upstream/GCA\_900206355.1\_upstream\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/fossil\_metagenome/latest\_assembly\_versions/GCA\_000208245.1\_ASM20824v1/GCA\_000208245.1\_ASM20824v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/fossil\_metagenome/latest\_assembly\_versions/GCA\_000208225.1\_ASM20822v1/GCA\_000208225.1\_ASM20822v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/plant\_metagenome/latest\_assembly\_versions/GCA\_900706555.1\_ORDI\_ANDO2/GCA\_900706555.1\_ORDI\_ANDO2\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/phyllosphere\_metagenome/latest\_assembly\_versions/GCA\_011800935.1\_ASM1180093v1/GCA\_011800935.1\_ASM1180093v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/permafrost\_metagenome/latest\_assembly\_versions/GCA\_013332135.1\_ASM1333213v1/GCA\_013332135.1\_ASM1333213v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/oil\_field\_metagenome/latest\_assembly\_versions/GCA\_001684095.1\_ASM168409v1/GCA\_001684095.1\_ASM168409v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/museum\_specimen\_metagenome/latest\_assembly\_versions/GCA\_002003105.1\_ASM200310v1/GCA\_002003105.1\_ASM200310v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/microbial\_mat\_metagenome/latest\_assembly\_versions/GCA\_000342605.1\_2865\_Mira-Geneious-final/GCA\_000342605.1\_2865\_Mira-Geneious-final\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/landfill\_metagenome/all\_assembly\_versions/GCA\_014872935.1\_ASM1487293v1/GCA\_014872935.1\_ASM1487293v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/landfill\_metagenome/all\_assembly\_versions/GCA\_014872925.1\_ASM1487292v1/GCA\_014872925.1\_ASM1487292v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/hydrocarbon\_metagenome/latest\_assembly\_versions/GCA\_001483165.1\_ASM148316v1/GCA\_001483165.1\_ASM148316v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/hypersaline\_lake\_metagenome/latest\_assembly\_versions/GCA\_001684355.1\_ASM168435v1/GCA\_001684355.1\_ASM168435v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/cold\_seep\_metagenome/latest\_assembly\_versions/GCA\_006844845.1\_ASM684484v1/GCA\_006844845.1\_ASM684484v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/estuary\_metagenome/latest\_assembly\_versions/GCA\_003226505.1\_ASM322650v1/GCA\_003226505.1\_ASM322650v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/marine\_sediment\_metagenome/latest\_assembly\_versions/GCA\_001045045.1\_MGS-ANC\_UA\_1/GCA\_001045045.1\_MGS-ANC\_UA\_1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/marine\_sediment\_metagenome/latest\_assembly\_versions/GCA\_001469255.1\_Aqaba\_Gulf/GCA\_001469255.1\_Aqaba\_Gulf\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/metagenomes/epibiont\_metagenome/latest\_assembly\_versions/GCA\_000206165.1\_ASM20616v1/GCA\_000206165.1\_ASM20616v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/all\_assembly\_versions/GCA\_900214915.1\_SRR5275893/GCA\_900214915.1\_SRR5275893\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/other/mixed\_sample/all\_assembly\_versions/GCA\_002754765.1\_ASM275476v1/GCA\_002754765.1\_ASM275476v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_004172875.1\_ASM417287v1/GCA\_004172875.1\_ASM417287v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_004173145.1\_ASM417314v1/GCA\_004173145.1\_ASM417314v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009908905.1\_ASM990890v1/GCA\_009908905.1\_ASM990890v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009909085.1\_ASM990908v1/GCA\_009909085.1\_ASM990908v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009993045.1\_ASM999304v1/GCA\_009993045.1\_ASM999304v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009993545.1\_ASM999354v1/GCA\_009993545.1\_ASM999354v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009993925.1\_ASM999392v1/GCA\_009993925.1\_ASM999392v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009994695.1\_ASM999469v1/GCA\_009994695.1\_ASM999469v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_009995505.1\_ASM999550v1/GCA\_009995505.1\_ASM999550v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_011088145.1\_ASM1108814v1/GCA\_011088145.1\_ASM1108814v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013203655.1\_ASM1320365v1/GCA\_013203655.1\_ASM1320365v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013205255.1\_ASM1320525v1/GCA\_013205255.1\_ASM1320525v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013205255.1\_ASM1320525v1/GCA\_013205255.1\_ASM1320525v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

genomobtained = await fetch("ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/unknown/taxid\_0/all\_assembly\_versions/GCA\_013205255.1\_ASM1320525v1/GCA\_013205255.1\_ASM1320525v1\_genomic.fna.gz");

load = await genomobtained.blob();

prokaryotic\_genomes.push(load);

}

}

async function symbiogenetic\_genomes() {

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/000/986/845/GCA\_000986845.1\_ASM98684v1/GCA\_000986845.1\_ASM98684v1\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = mitochondrial\_genome;

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/016/183/945/GCA\_016183945.1\_ASM1618394v1/GCA\_016183945.1\_ASM1618394v1\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = mitochondrial\_genome;

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/595/GCF\_000002595.2\_Chlamydomonas\_reinhardtii\_v5.5/GCF\_000002595.2\_Chlamydomonas\_reinhardtii\_v5.5\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/870/225/GCF\_001870225.1\_ASM187022v1/GCF\_001870225.1\_ASM187022v1\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/002/595/GCF\_000002595.2\_Chlamydomonas\_reinhardtii\_v5.5/GCF\_000002595.2\_Chlamydomonas\_reinhardtii\_v5.5\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/146/045/GCF\_000146045.2\_R64/GCF\_000146045.2\_R64\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/731/375/GCA\_009731375.1\_PmicroMYN1\_1.0/GCA\_009731375.1\_PmicroMYN1\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/002/378/185/GCA\_002378185.1\_ASM237818v1/GCA\_002378185.1\_ASM237818v1\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

genomobtained = await fetch("https://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/009/731/375/GCA\_009731375.1\_PmicroMYN1\_1.0/GCA\_009731375.1\_PmicroMYN1\_1.0\_genomic.fna.gz");

load = await genomobtained.blob();

symvolgen.push(load);

}

//unfinished functions for connectome data

function fly\_connectome() {

var x = 0;

}

function vertebrate\_connectome() {

var x = 0;

}

function terrain() {

unused\_variable = 5;

if (unused\_variable == 5) {

used\_radius\_layer\_earth = 3.948516444e+41;

used\_density\_parameter = 1.474334865e+25;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

material\_temperature\_\_kelvins = 273.18;

for (var count205 = 0; count205 < generation\_parameter\_3; count205++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count206 = 0; count206 < generation\_parameter\_3; count206++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count207 = 0; count207 < generation\_parameter\_3; count207++) {

cursor1z = cursor1z + used\_density\_parameter;

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1x - planet\_x;

relative\_z = cursor1x - planet\_x;

if (Math.abs(relative\_x) <= 5) {

relative\_x = 6;

}

if (Math.abs(relative\_y) <= 5) {

relative\_y = 6;

}

if (Math.abs(relative\_z) <= 5) {

relative\_z = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_x / relative\_y);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))));

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9497172158) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41 + 2.351412394e+37 \* ((0.008 - Math.abs(((lunar\_eta\_coordinate \* (180 / Math.PI) - 86.9497172158) - 0) - 0.008)) / 0.008)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

innitiate\_terrain\_crust();

}

}

}

}

} else {

terrain\_crust\_2();

}

} else {

terrain\_crust\_2();

}

} else {

terrain\_crust\_2();

}

} else {

terrain\_crust\_2();

}

}

cursor1z = cursor1x - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

soil();

}

function terrain\_crust\_2() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.860685692) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -20.8724751657) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9636119527) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.951822479) {

if (0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684) > 0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) {

innitiate\_terrain\_crust();

}

}

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) {

innitiate\_terrain\_crust();

}

}

}

}

}

} else {

terrain\_crust\_3();

}

} else {

terrain\_crust\_3();

}

} else {

terrain\_crust\_3();

}

} else {

terrain\_crust\_3();

}

}

function terrain\_crust\_3() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 89.12348421518035) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41) {

innitiate\_terrain\_crust();

}

}

} else {

terrain\_crust\_4();

}

} else {

terrain\_crust\_4();

}

} else {

terrain\_crust\_4();

}

} else {

terrain\_crust\_4();

}

}

function terrain\_crust\_4() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9497172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41) {

innitiate\_terrain\_crust();

}

}

} else {

terrain\_crust\_5();

}

} else {

terrain\_crust\_5();

}

} else {

terrain\_crust\_5();

}

} else {

terrain\_crust\_5();

}

}

function terrain\_crust\_5() {

obtain\_altitude();

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < altitude) {

innitiate\_terrain\_crust();

}

}

}

}

function innitiate\_terrain\_crust() {

set\_earth\_rotation();

randomize\_magma\_atom();

}

//unfinished function, originally intended to fetch altitude data from Google Earth.

function obtain\_altitude() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -70.4638506) {

altitude = 3.942328517e+41 + (8849 \* 6.187927353e+34) \* Math.abs(Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 10) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 10) / 180 \* Math.PI));

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -60.4638506) {

altitude = 3.942328517e+41 + (5000 \* 6.187927353e+34) \* Math.abs(Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 20) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 20) / 180 \* Math.PI));

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -50.4638506) {

altitude = 3.942328517e+41 + (3000 \* 6.187927353e+34) \* Math.abs(Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 25) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 25) / 180 \* Math.PI));

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -21.4638506) {

altitude = 3.942328517e+41 + Math.abs((((Math.abs(lunar\_phi\_coordinate \* (180 / Math.PI) - 90) - 21.4638506) / 70) \* (7000 \* 6.187927353e+34)) \* (Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 25) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 25) / 180 \* Math.PI)));

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8118435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

altitude = 3.942328517e+41 + 2 \* 6.187927353e+34;

} else {

altitude = 3.942328517e+41 - 100 \* 6.187927353e+34;

}

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -15.8118435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

altitude = 3.942328517e+41 + Math.abs((1000 \* 6.187927353e+34) \* (Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 100) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 100) / 180 \* Math.PI)));

} else if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 87.9657172158) {

altitude = 3.942328517e+41 + (1 \* 6.187927353e+34 + (1000 \* 6.187927353e+34) \* (Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 100) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 100) / 180 \* Math.PI)));

} else {

altitude = 3.942328517e+41 - 1000 \* 6.187927353e+34;

}

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -10.8118435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 40) {

altitude = (3.942328517e+41 - 500 \* 6.187927353e+34) + Math.abs((((Math.abs(lunar\_phi\_coordinate \* (180 / Math.PI) - 90) - 1 \* -10.8118435868) / 10) \* (1000 \* 6.187927353e+34)) \* (Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 25) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 25) / 180 \* Math.PI)));

} else if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

altitude = 3.942328517e+41 + (1 \* 6.187927353e+34 + (1000 \* 6.187927353e+34) \* (Math.sin(((lunar\_phi\_coordinate \* (180 / Math.PI)) \* 100) / 180 \* Math.PI) \* Math.sin(((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) \* 100) / 180 \* Math.PI)));

} else {

altitude = 3.942328517e+41 - 2000 \* 6.187927353e+34;

}

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < 20) {

altitude = 3.942328517e+41 - 3000 \* 6.187927353e+34;

} else if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < 50) {

altitude = 3.942328517e+41 - 5000 \* 6.187927353e+34;

} else {

altitude = 3.942328517e+41 - 11034 \* 6.187927353e+34;

}

}

function soil() {

unused\_variable = 5;

if (unused\_variable == 5) {

used\_radius\_layer\_earth = 3.948516444e+41;

used\_density\_parameter = 1.474334865e+25;

centerpoint\_x = planet\_x - used\_radius\_layer\_earth;

centerpoint\_y = planet\_y - used\_radius\_layer\_earth;

centerpoint\_z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

material\_temperature\_\_kelvins = 293.18 + math\_random\_int(1, 1e+100) / 1e+100;

for (var count216 = 0; count216 < generation\_parameter\_3; count216++) {

centerpoint\_x = centerpoint\_x + used\_density\_parameter;

for (var count217 = 0; count217 < generation\_parameter\_3; count217++) {

centerpoint\_y = centerpoint\_y + used\_density\_parameter;

for (var count218 = 0; count218 < generation\_parameter\_3; count218++) {

centerpoint\_z = centerpoint\_z + used\_density\_parameter;

relative\_x = centerpoint\_x - planet\_x;

relative\_y = centerpoint\_x - planet\_x;

relative\_z = centerpoint\_x - planet\_x;

if (Math.abs(relative\_x) <= 5) {

relative\_x = 6;

}

if (Math.abs(relative\_y) <= 5) {

relative\_y = 6;

}

if (Math.abs(relative\_z) <= 5) {

relative\_z = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_x / relative\_y);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))));

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9497172158) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 + 1.23758547e+32) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41 + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < (3.942266637e+41 + 2.351412394e+37 \* ((0.008 - Math.abs(((lunar\_eta\_coordinate \* (180 / Math.PI) - 86.9497172158) - 0) - 0.008)) / 0.008)) + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942266637e+41 + 2.351412394e+37 \* ((0.008 - Math.abs(((lunar\_eta\_coordinate \* (180 / Math.PI) - 86.9497172158) - 0) - 0.008)) / 0.008)) + 1.23758547e+32) {

set\_earth\_rotation();

randomize\_soil();

}

}

}

}

}

} else {

soil\_2();

}

} else {

soil\_2();

}

} else {

soil\_2();

}

} else {

soil\_2();

}

}

centerpoint\_z = centerpoint\_x - generation\_parameter\_3 \* used\_density\_parameter;

}

centerpoint\_y = centerpoint\_y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

hydrosphere();

}

function soil\_2() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.860685692) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -20.8724751657) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9636119527) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.951822479) {

if (0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684) > 0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 + 1.23758547e+32) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41 + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 1.23758547e+32) {

set\_earth\_rotation();

randomize\_soil();

}

}

}

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 + 1.23758547e+32) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41 + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 6.175551499e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 1.23758547e+32) {

randomize\_soil();

set\_earth\_rotation();

}

}

}

}

}

}

} else {

soil\_3();

}

} else {

soil\_3();

}

} else {

soil\_3();

}

} else {

soil\_3();

}

}

function soil\_3() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 89.12348421518035) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41) {

set\_earth\_rotation();

randomize\_soil();

}

}

} else {

soil\_4();

}

} else {

soil\_4();

}

} else {

soil\_4();

}

} else {

soil\_4();

}

}

function soil\_4() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9497172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41) {

set\_earth\_rotation();

randomize\_soil();

}

}

} else {

soil\_5();

}

} else {

soil\_5();

}

} else {

soil\_5();

}

} else {

soil\_5();

}

}

function soil\_5() {

obtain\_altitude();

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 + 1.23758547e+32) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > altitude + 1.23758547e+32) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < altitude + 6.175551499e+34) {

set\_earth\_rotation();

randomize\_soil();

}

}

}

}

}

function randomize\_soil() {

hollow\_soil();

randomizer = math\_random\_int(1, 1e+100);

unused\_variable = 5;

if (randomizer <= (1 / (1000000000 \* 10000)) \* 1e+100) {

ploidy = 2;

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

randomize\_microbial\_genome();

supported\_cell();

pelagibacter\_genome2();

spawnerz = spawnerz + 6.187927353e+29;

supporting\_cell();

} else if (randomizer <= (1 / (1000000000 \* 10)) \* 1e+100) {

ploidy = 2;

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

randomize\_eukaryotic\_genome();

omnicyte();

} else if (randomizer <= (1 / 1000000000) \* 1e+100) {

randomizer = math\_random\_int(1, 100);

ploidy = 2;

sizemax = 6.002289533e+31;

if (randomizer <= 10) {

nematode\_genome2();

invertebrate\_female();

} else if (randomizer <= 20) {

nematode\_genome2();

invertebrate\_male();

} else if (randomizer <= 30) {

tardigrade\_genome2();

invertebrate\_female();

} else if (randomizer <= 40) {

tardigrade\_genome2();

invertebrate\_male();

} else if (randomizer <= 50) {

rotifer\_genome2();

invertebrate\_male();

} else {

rotifer\_genome2();

invertebrate\_female();

}

} else if (randomizer <= 0.03 \* 1e+100) {

soil\_organic\_matter\_2();

} else if (randomizer <= 0.05 \* 1e+100) {

soil\_organic\_matter\_1();

} else if (randomizer <= 0.3 \* 1e+100) {

soil\_droplet();

} else {

sand\_grain();

}

}

function hollow\_soil() {

if (list\_\_universe\_string\_entities\_\_\_default.length > 2) {

l1 = list\_\_universe\_string\_entities\_\_\_default.length;

c1 = 0;

for (var count219 = 0; count219 < l1; count219++) {

if (c1 + 2 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(c1 + 1) - 1];

c1 = c1 + 1;

obtain\_string\_location();

if (string\_y < centerpoint\_y) {

if (string\_y < centerpoint\_y - 6.187927353e+31) {

if (string\_x < centerpoint\_x) {

if (string\_x < centerpoint\_x - 6.187927353e+31) {

if (string\_z > centerpoint\_z) {

if (string\_z < centerpoint\_z + 6.187927353e+31) {

list.splice(c1 - 1, 1);

c1 = c1 - 1;

}

}

}

}

}

}

}

}

}

}

function sand\_grain() {

if (unused\_variable == 5) {

guider\_x = centerpoint\_x - 6.187927353e+31 / 2;

guider\_y = centerpoint\_y - 6.187927353e+31 / 2;

guider\_z = centerpoint\_z + 6.187927353e+31 / 2;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count220 = 0; count220 < generation\_parameter\_x; count220++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count221 = 0; count221 < generation\_parameter\_y; count221++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count222 = 0; count222 < generation\_parameter\_z; count222++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count223 = 0; count223 < generation\_parameter\_x; count223++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count224 = 0; count224 < generation\_parameter\_y; count224++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count225 = 0; count225 < generation\_parameter\_z; count225++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count226 = 0; count226 < generation\_parameter\_x; count226++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count227 = 0; count227 < generation\_parameter\_y; count227++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count228 = 0; count228 < generation\_parameter\_z; count228++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count229 = 0; count229 < generation\_parameter\_x; count229++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count230 = 0; count230 < generation\_parameter\_y; count230++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count231 = 0; count231 < generation\_parameter\_z; count231++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count232 = 0; count232 < generation\_parameter\_x; count232++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count233 = 0; count233 < generation\_parameter\_y; count233++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count234 = 0; count234 < generation\_parameter\_z; count234++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count235 = 0; count235 < generation\_parameter\_x; count235++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count236 = 0; count236 < generation\_parameter\_y; count236++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count237 = 0; count237 < generation\_parameter\_z; count237++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count238 = 0; count238 < generation\_parameter\_x; count238++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count239 = 0; count239 < generation\_parameter\_y; count239++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count240 = 0; count240 < generation\_parameter\_z; count240++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count241 = 0; count241 < generation\_parameter\_x; count241++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count242 = 0; count242 < generation\_parameter\_y; count242++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count243 = 0; count243 < generation\_parameter\_z; count243++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 1.474334865e+25;

density\_parameter\_y = 1.474334865e+25;

density\_parameter\_z = 1.474334865e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x - 4);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y - 4);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z - 4);

for (var count244 = 0; count244 < generation\_parameter\_x; count244++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count245 = 0; count245 < generation\_parameter\_y; count245++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count246 = 0; count246 < generation\_parameter\_z; count246++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) > 6.187927353e+31) {

set\_earth\_rotation();

randomize\_air\_molecule();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function soil\_droplet() {

if (unused\_variable == 5) {

guider\_x = centerpoint\_x - 6.187927353e+31 / 2;

guider\_y = centerpoint\_y - 6.187927353e+31 / 2;

guider\_z = centerpoint\_z + 6.187927353e+31 / 2;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count247 = 0; count247 < generation\_parameter\_x; count247++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count248 = 0; count248 < generation\_parameter\_y; count248++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count249 = 0; count249 < generation\_parameter\_z; count249++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 1.474334865e+25;

density\_parameter\_y = 1.474334865e+25;

density\_parameter\_z = 1.474334865e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x - 4);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y - 4);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z - 4);

for (var count250 = 0; count250 < generation\_parameter\_x; count250++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count251 = 0; count251 < generation\_parameter\_y; count251++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count252 = 0; count252 < generation\_parameter\_z; count252++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) > 6.187927353e+31) {

set\_earth\_rotation();

randomize\_air\_molecule();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function soil\_organic\_matter\_1() {

if (unused\_variable == 5) {

guider\_x = centerpoint\_x - 6.187927353e+31 / 2;

guider\_y = centerpoint\_y - 6.187927353e+31 / 2;

guider\_z = centerpoint\_z + 6.187927353e+31 / 2;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count253 = 0; count253 < generation\_parameter\_x; count253++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count254 = 0; count254 < generation\_parameter\_y; count254++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count255 = 0; count255 < generation\_parameter\_z; count255++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

randomize\_nutrient\_1();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 1.474334865e+25;

density\_parameter\_y = 1.474334865e+25;

density\_parameter\_z = 1.474334865e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x - 4);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y - 4);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z - 4);

for (var count256 = 0; count256 < generation\_parameter\_x; count256++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count257 = 0; count257 < generation\_parameter\_y; count257++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count258 = 0; count258 < generation\_parameter\_z; count258++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) > 6.187927353e+31) {

set\_earth\_rotation();

randomize\_air\_molecule();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function soil\_organic\_matter\_2() {

if (unused\_variable == 5) {

guider\_x = centerpoint\_x - 6.187927353e+31 / 2;

guider\_y = centerpoint\_y - 6.187927353e+31 / 2;

guider\_z = centerpoint\_z + 6.187927353e+31 / 2;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z);

for (var count259 = 0; count259 < generation\_parameter\_x; count259++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count260 = 0; count260 < generation\_parameter\_y; count260++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count261 = 0; count261 < generation\_parameter\_z; count261++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < 6.187908789e+31) {

set\_earth\_rotation();

randomize\_nutrient\_3();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 0.75;

cursor1y = cursor1y + density\_parameter\_x / 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 1.474334865e+25;

density\_parameter\_y = 1.474334865e+25;

density\_parameter\_z = 1.474334865e+25;

generation\_parameter\_x = Math.round(6.187927353e+31 / density\_parameter\_x - 4);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y - 4);

generation\_parameter\_z = Math.round(6.187927353e+31 / density\_parameter\_z - 4);

for (var count262 = 0; count262 < generation\_parameter\_x; count262++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count263 = 0; count263 < generation\_parameter\_y; count263++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count264 = 0; count264 < generation\_parameter\_z; count264++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) > 6.187927353e+31) {

set\_earth\_rotation();

randomize\_air\_molecule();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function sort\_allotrope\_bar() {

if (sorter == 11) {

unused\_variable = 5;

} else if (sorter == 12) {

graphene\_1();

} else if (sorter == 13) {

graphene\_2();

} else if (sorter == 14) {

diamond\_4();

} else if (sorter == 15) {

diamond\_3();

}

}

function hydrosphere() {

unused\_variable = 5;

if (unused\_variable == 5) {

used\_radius\_layer\_earth = 3.948516444e+41;

used\_density\_parameter = 1.922068735e+25;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

material\_temperature\_\_kelvins = 302.18 + math\_random\_int(1, 1e+31) / 1e+31;

for (var count265 = 0; count265 < generation\_parameter\_3; count265++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count266 = 0; count266 < generation\_parameter\_3; count266++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count267 = 0; count267 < generation\_parameter\_3; count267++) {

cursor1z = cursor1z + used\_density\_parameter;

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1x - planet\_x;

relative\_z = cursor1x - planet\_x;

if (Math.abs(relative\_x) <= 5) {

relative\_x = 6;

}

if (Math.abs(relative\_y) <= 5) {

relative\_y = 6;

}

if (Math.abs(relative\_z) <= 5) {

relative\_z = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_x / relative\_y);

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))));

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9636119526) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.951822479) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942266637e+41 + 2.351412394e+37 \* ((0.008 - Math.abs(((lunar\_eta\_coordinate \* (180 / Math.PI) - 86.9497172158) - 0) - 0.008)) / 0.008)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

}

}

} else {

hydrosphere\_2();

}

} else {

hydrosphere\_2();

}

} else {

hydrosphere\_2();

}

} else {

hydrosphere\_2();

}

}

cursor1z = cursor1x - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

atmosphere();

}

function hydrosphere\_2() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.860685692) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -20.8724751657) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9636119527) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.951822479) {

if (0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684) > 0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.933046626e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.94234708e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

}

}

}

} else {

hydrosphere\_3();

}

} else {

hydrosphere\_3();

}

} else {

hydrosphere\_3();

}

} else {

hydrosphere\_3();

}

}

function hydrosphere\_3() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9636119526) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 89.12348421518035) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942266637e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 - (Math.abs((lunar\_eta\_coordinate \* (180 / Math.PI) - 180) - 86.9636119526) / 95) \* (6.187927353e+37 / 2)) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

}

} else {

hydrosphere\_4();

}

} else {

hydrosphere\_4();

}

} else {

hydrosphere\_4();

}

} else {

hydrosphere\_4();

}

}

function hydrosphere\_4() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.951822479) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942266637e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 - (Math.abs(86.951822479 - (lunar\_eta\_coordinate \* (180 / Math.PI) - 180)) / 95) \* (6.187927353e+37 / 2)) {

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

}

}

} else {

hydrosphere\_5();

}

} else {

hydrosphere\_5();

}

} else {

hydrosphere\_5();

}

} else {

hydrosphere\_5();

}

}

function hydrosphere\_5() {

obtain\_altitude();

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41 - Math.abs(lunar\_phi\_coordinate \* (180 / Math.PI) - 90) \* (1.23758547e+37 / 90)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > altitude) {

set\_earth\_rotation();

randomize\_seawater\_molecule();

}

}

}

}

function atmosphere() {

unused\_variable = 5;

if (unused\_variable == 5) {

used\_radius\_layer\_earth = 4.00420779e+41;

used\_density\_parameter = 2.115113925e+26;

cursor1x = planet\_x - used\_radius\_layer\_earth;

cursor1y = planet\_y - used\_radius\_layer\_earth;

cursor1z = planet\_z - used\_radius\_layer\_earth;

generation\_parameter\_3 = Math.round((used\_radius\_layer\_earth \* 2) / used\_density\_parameter);

material\_temperature\_\_kelvins = 301.18 + math\_random\_int(1, 3e+40) / 1e+40;

for (var count268 = 0; count268 < generation\_parameter\_3; count268++) {

cursor1x = cursor1x + used\_density\_parameter;

for (var count269 = 0; count269 < generation\_parameter\_3; count269++) {

cursor1y = cursor1y + used\_density\_parameter;

for (var count270 = 0; count270 < generation\_parameter\_3; count270++) {

cursor1z = cursor1z + used\_density\_parameter;

relative\_x = cursor1x - planet\_x;

relative\_y = cursor1x - planet\_x;

relative\_z = cursor1x - planet\_x;

if (Math.abs(relative\_x) <= 5) {

relative\_x = 6;

}

if (Math.abs(relative\_y) <= 5) {

relative\_y = 6;

}

if (Math.abs(relative\_z) <= 5) {

relative\_z = 6;

}

lunar\_eta\_coordinate = Math.atan(relative\_x / relative\_y) / Math.PI \* 180;

if (relative\_y < 0) {

lunar\_eta\_coordinate = Math.PI + (Math.PI + lunar\_eta\_coordinate);

}

lunar\_phi\_coordinate = Math.acos(relative\_z / Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2))))) / Math.PI \* 180;

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9497172158) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942266637e+41 + 2.351412394e+37 \* ((0.008 - Math.abs(((lunar\_eta\_coordinate \* (180 / Math.PI) - 86.9497172158) - 0) - 0.008)) / 0.008)) + 6.187927353e+34) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

}

} else {

atmosphere\_2();

}

} else {

atmosphere\_2();

}

} else {

atmosphere\_2();

}

} else {

atmosphere\_2();

}

}

cursor1z = cursor1x - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y - generation\_parameter\_3 \* used\_density\_parameter;

}

}

flora();

}

function atmosphere\_2() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.860685692) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -20.8724751657) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9636119527) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.951822479) {

if (0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684) > 0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_eta\_coordinate) - Math.abs(86.951822479)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 6.187927353e+34) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > (3.942328517e+41 + ((0.00589473684 - Math.abs((Math.abs(lunar\_phi\_coordinate) - Math.abs(-20.860685692)) - 0.00589473684)) / 0.00589473684) \* 1.732619659e+37) + 6.187927353e+34) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

}

}

} else {

atmosphere\_3();

}

} else {

atmosphere\_3();

}

} else {

atmosphere\_3();

}

} else {

atmosphere\_3();

}

}

function atmosphere\_3() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.9657172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 89.12348421518035) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

} else {

atmosphere\_4();

}

} else {

atmosphere\_4();

}

} else {

atmosphere\_4();

}

} else {

atmosphere\_4();

}

}

function atmosphere\_4() {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 > -21.4638506) {

if (lunar\_phi\_coordinate \* (180 / Math.PI) - 90 < -20.8718435868) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 < 86.9497172158) {

if (lunar\_eta\_coordinate \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

} else {

atmosphere\_5();

}

} else {

atmosphere\_5();

}

} else {

atmosphere\_5();

}

} else {

atmosphere\_5();

}

}

function atmosphere\_5() {

obtain\_altitude();

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.942328517e+41 - Math.abs(lunar\_phi\_coordinate \* (180 / Math.PI) - 90) \* (1.23758547e+37 / 90)) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > altitude + 6.187927353e+34) {

set\_earth\_rotation();

innitiate\_atmosphere();

}

}

}

}

function innitiate\_atmosphere() {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.942328517e+41) {

randomize\_air\_molecule();

} else if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) > 3.951610408e+41) {

if (Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) < 3.963986262e+41) {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= (1 / Math.pow(2, Math.abs(Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) - 3.942328517e+41) / 3.465239318e+38)) \* 1e+30) {

randomize\_ozone\_layer();

}

} else {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= (1 / Math.pow(2, Math.abs(Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) - 3.942328517e+41) / 3.465239318e+38)) \* 1e+30) {

randomize\_air\_molecule();

}

}

} else {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= (1 / Math.pow(2, Math.abs(Math.sqrt(Math.abs(Math.pow(relative\_x, 2) + (Math.pow(relative\_y, 2) + Math.pow(relative\_z, 2)))) - 3.942328517e+41) / 3.465239318e+38)) \* 1e+30) {

randomize\_air\_molecule();

}

}

}

function randomize\_air\_molecule() {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer < (0.4 / 100) \* 1e+30) {

water\_molecule();

} else {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer < (1 / 20000000) \* 1e+30) {

air\_xenon();

} else if (randomizer < (0.000055 / 100 + 1 / 20000000) \* 1e+30) {

air\_hydrogen();

} else if (randomizer < (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000)) \* 1e+30) {

air\_krypton();

} else if (randomizer < (0.000524 / 100 + (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000))) \* 1e+30) {

air\_helium();

} else if (randomizer < (0.001818 / 100 + (0.000524 / 100 + (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000)))) \* 1e+30) {

air\_neon();

} else if (randomizer < (0.0407 / 100 + (0.001818 / 100 + (0.000524 / 100 + (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000))))) \* 1e+30) {

carbon\_dioxide();

} else if (randomizer < (0.934 / 100 + (0.0407 / 100 + (0.001818 / 100 + (0.000524 / 100 + (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000)))))) \* 1e+30) {

air\_argon();

} else if (randomizer < (20.946 / 100 + (0.934 / 100 + (0.0407 / 100 + (0.001818 / 100 + (0.000524 / 100 + (0.000114 / 100 + (0.000055 / 100 + 1 / 20000000))))))) \* 1e+30) {

air\_oxygen();

} else {

air\_nitrogen();

}

}

}

function randomize\_ozone\_layer() {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer < (1 / 5500000) \* 1e+30) {

ozone();

} else {

randomize\_air\_molecule();

}

}

function randomize\_freshwater\_molecule() {

randomizer = math\_random\_int(1, 1e+100);

if (randomizer <= 0.00000488977 \* 1e+100) {

air\_oxygen();

} else if (randomizer <= (0.00000849729 + 0.00000488977) \* 1e+100) {

air\_nitrogen();

} else if (randomizer <= (0.00001377706 / 53.6) \* 1e+100) {

salt\_iodine();

} else if (randomizer <= ((0.00025 + 0.00001377706) / 53.6) \* 1e+100) {

bicarbonate();

} else if (randomizer <= ((0.000142 + (0.00025 + 0.00001377706)) / 53.6) \* 1e+100) {

silicon\_unbound\_();

} else if (randomizer <= ((0.009719 + (0.000142 + (0.00025 + 0.00001377706))) / 53.6) \* 1e+100) {

salt\_potassium();

} else if (randomizer <= ((0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706)))) / 53.6) \* 1e+100) {

salt\_calcium();

} else if (randomizer <= ((0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706)))) / 53.6 + 3 / 1000000) \* 1e+100) {

salt\_magnesium();

} else if (randomizer <= ((0.154 + (0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706))))) / 53.6 + 3 / 1000000) \* 1e+100) {

salt\_sodium();

} else if (randomizer <= ((0.154 + (0.154 + (0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706)))))) / 53.6 + 3 / 1000000) \* 1e+100) {

salt\_chloride();

} else {

water\_molecule();

}

}

function randomize\_seawater\_molecule() {

randomizer = math\_random\_int(1, 1e+100);

if (randomizer <= 0.00000488977 \* 1e+100) {

air\_oxygen();

} else if (randomizer <= (0.00000849729 + 0.00000488977) \* 1e+100) {

air\_nitrogen();

} else if (randomizer <= (0.00001377706 / 53.6) \* 1e+100) {

salt\_iodine();

} else if (randomizer <= ((0.00025 + 0.00001377706) / 53.6) \* 1e+100) {

bicarbonate();

} else if (randomizer <= ((0.000142 + (0.00025 + 0.00001377706)) / 53.6) \* 1e+100) {

silicon\_unbound\_();

} else if (randomizer <= ((0.009719 + (0.000142 + (0.00025 + 0.00001377706))) / 53.6) \* 1e+100) {

salt\_potassium();

} else if (randomizer <= ((0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706)))) / 53.6) \* 1e+100) {

salt\_calcium();

} else if (randomizer <= ((0.052346 + (0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706))))) / 53.6) \* 1e+100) {

salt\_magnesium();

} else if (randomizer <= ((0.459174 + (0.052346 + (0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706)))))) / 53.6) \* 1e+100) {

salt\_sodium();

} else if (randomizer <= ((0.536158 + (0.459174 + (0.052346 + (0.01 + (0.009719 + (0.000142 + (0.00025 + 0.00001377706))))))) / 53.6) \* 1e+100) {

salt\_chloride();

} else {

water\_molecule();

}

}

function callus\_medium() {

randomizer = math\_random\_int(1, 1000000);

if (randomizer <= 2000) {

auxin();

} else if (randomizer <= 2020) {

cytokinin();

} else {

randomize\_ECF();

}

}

function root\_medium() {

randomizer = math\_random\_int(1, 1000000);

if (randomizer <= 2000) {

auxin();

} else if (randomizer <= 2200) {

cytokinin();

} else {

randomize\_ECF();

}

}

function shoot\_medium() {

randomizer = math\_random\_int(1, 1000000);

if (randomizer <= 20) {

auxin();

} else if (randomizer <= 1020) {

cytokinin();

} else {

randomize\_ECF();

}

}

function randomize\_ECF() {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= 0.00000488977 \* 1e+30) {

air\_oxygen();

} else if (randomizer <= (0.00000488977 + 0.00000849729) \* 1e+30) {

air\_nitrogen();

} else {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= ((1 / 1000) / 55.346) \* 1e+30) {

ferrosulphur();

} else if (randomizer <= ((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30) {

sulphate();

} else if (randomizer <= (4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)) {

molybdate();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))) {

salt\_lithium();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))) {

salt\_manganese();

} else if (randomizer <= (8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))) {

salt\_chromium();

} else if (randomizer <= (0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))) {

borate();

} else if (randomizer <= (7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))) {

salt\_iodine();

} else if (randomizer <= (0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))) {

salt\_copper();

} else if (randomizer <= (0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))) {

salt\_zinc();

} else if (randomizer <= ((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))) {

salt\_calcium();

} else if (randomizer <= ((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))) {

salt\_magnesium();

} else if (randomizer <= ((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))) {

bicarbonate();

} else if (randomizer <= ((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))))) {

salt\_chloride();

} else if (randomizer <= ((145 / 1000) / 55.346) \* 1e+30 + (((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))))) {

salt\_sodium();

} else if (randomizer <= ((4 / 1000) / 55.346) \* 1e+30 + (((145 / 1000) / 55.346) \* 1e+30 + (((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))))))) {

salt\_potassium();

} else {

water\_molecule();

}

}

}

function randomize\_cytosol() {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= 0.00000488977 \* 1e+30) {

air\_oxygen();

} else if (randomizer <= (0.00000488977 + 0.00000849729) \* 1e+30) {

air\_nitrogen();

} else {

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= ((1 / 1000) / 55.346) \* 1e+30) {

ferrosulphur();

} else if (randomizer <= ((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30) {

sulphate();

} else if (randomizer <= (4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)) {

molybdate();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))) {

salt\_lithium();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))) {

salt\_manganese();

} else if (randomizer <= (8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))) {

salt\_chromium();

} else if (randomizer <= (0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))) {

borate();

} else if (randomizer <= (7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))) {

salt\_iodine();

} else if (randomizer <= (0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))) {

salt\_copper();

} else if (randomizer <= (0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))) {

salt\_zinc();

} else if (randomizer <= ((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))) {

salt\_calcium();

} else if (randomizer <= ((0.8 / 1000) / 55.346) \* 1e+30 + (((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))) {

salt\_magnesium();

} else if (randomizer <= ((12 / 1000) / 55.346) \* 1e+30 + (((0.8 / 1000) / 55.346) \* 1e+30 + (((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))) {

bicarbonate();

} else if (randomizer <= ((4 / 1000) / 55.346) \* 1e+30 + (((12 / 1000) / 55.346) \* 1e+30 + (((0.8 / 1000) / 55.346) \* 1e+30 + (((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))))) {

salt\_chloride();

} else if (randomizer <= ((12 / 1000) / 55.346) \* 1e+30 + (((4 / 1000) / 55.346) \* 1e+30 + (((12 / 1000) / 55.346) \* 1e+30 + (((0.8 / 1000) / 55.346) \* 1e+30 + (((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))))) {

salt\_sodium();

} else if (randomizer <= ((139 / 1000) / 55.346) \* 1e+30 + (((12 / 1000) / 55.346) \* 1e+30 + (((4 / 1000) / 55.346) \* 1e+30 + (((12 / 1000) / 55.346) \* 1e+30 + (((0.8 / 1000) / 55.346) \* 1e+30 + (((0.00001 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))))))) {

salt\_potassium();

} else {

water\_molecule();

}

}

}

function randomize\_mineral() {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927353e+25, cursor1y - 6.187927353e+25, cursor1x - 6.187927353e+25];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, ((139 / 1000) / 55.346) \* 1e+30 + (((145 / 1000) / 55.346) \* 1e+30 + (((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))))));

if (randomizer <= ((1 / 1000) / 55.346) \* 1e+30) {

ferrosulphur();

} else if (randomizer <= ((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30) {

sulphate();

} else if (randomizer <= (4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)) {

molybdate();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))) {

salt\_lithium();

} else if (randomizer <= (0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))) {

salt\_manganese();

} else if (randomizer <= (8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))) {

salt\_chromium();

} else if (randomizer <= (0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))) {

borate();

} else if (randomizer <= (7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))) {

salt\_iodine();

} else if (randomizer <= (0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))) {

salt\_copper();

} else if (randomizer <= (0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))) {

salt\_zinc();

} else if (randomizer <= ((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))) {

salt\_calcium();

} else if (randomizer <= ((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))) {

salt\_magnesium();

} else if (randomizer <= ((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))) {

bicarbonate();

} else if (randomizer <= ((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30))))))))))))) {

salt\_chloride();

} else if (randomizer <= ((145 / 1000) / 55.346) \* 1e+30 + (((116 / 1000) / 55.346) \* 1e+30 + (((29 / 1000) / 55.346) \* 1e+30 + (((1.5 / 1000) / 55.346) \* 1e+30 + (((1.8 / 1000) / 55.346) \* 1e+30 + ((0.00031 / 3) \* 1e+30 + ((0.0000104 / 3) \* 1e+30 + ((7.5e-7 / 3) \* 1e+30 + ((0.000003 / 3) \* 1e+30 + ((8.9e-8 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((0.0000015 / 3) \* 1e+30 + ((4.5e-8 / 3) \* 1e+30 + (((0.3 / 1000) / 55.346) \* 1e+30 + ((1 / 1000) / 55.346) \* 1e+30)))))))))))))) {

salt\_sodium();

} else {

salt\_potassium();

}

list\_molecules.push(['mineral', cursor1z, cursor1y, cursor1x, cursor1z + 6.187927352e+25, cursor1y - 6.187927352e+25, cursor1x - 6.187927352e+25]);

}

}

function nutrient\_1(){

randomize\_nutrient\_1();

}

function nutrient\_2(){

randomize\_nutrient\_2();

}

function nutrient\_3(){

randomize\_nutrient\_3();

}

function randomize\_nutrient\_1() {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 300) {

randomize\_mineral();

} else if (randomizer <= 302) {

randomize\_vitamin();

} else {

randomize\_amino\_acid();

}

}

function randomize\_nutrient\_2() {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 300) {

randomize\_mineral();

} else if (randomizer <= 302) {

randomize\_vitamin();

} else {

glucose();

}

}

function randomize\_nutrient\_3() {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 1000) {

randomize\_mineral();

} else if (randomizer <= 1005) {

randomize\_vitamin();

} else {

randomize\_fat();

}

}

function randomize\_amino\_acid() {

randomizer = math\_random\_int(1, 10000000000);

if (randomizer <= 170) {

selenocysteine();

} else {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 876) {

peptidesequence = 'A';

} else if (randomizer <= 1454) {

peptidesequence = 'R';

} else if (randomizer <= 1847) {

peptidesequence = 'N';

} else if (randomizer <= 2396) {

peptidesequence = 'D';

} else if (randomizer <= 2534) {

peptidesequence = 'C';

} else if (randomizer <= 2924) {

peptidesequence = 'Q';

} else if (randomizer <= 3556) {

peptidesequence = 'E';

} else if (randomizer <= 4259) {

peptidesequence = 'G';

} else if (randomizer <= 4485) {

peptidesequence = 'H';

} else if (randomizer <= 5034) {

peptidesequence = 'I';

} else if (randomizer <= 6002) {

peptidesequence = 'L';

} else if (randomizer <= 6521) {

peptidesequence = 'K';

} else if (randomizer <= 6753) {

peptidesequence = 'M';

} else if (randomizer <= 7140) {

peptidesequence = 'F';

} else if (randomizer <= 7642) {

peptidesequence = 'P';

} else if (randomizer <= 8356) {

peptidesequence = 'S';

} else if (randomizer <= 8909) {

peptidesequence = 'T';

} else if (randomizer <= 9034) {

peptidesequence = 'W';

} else if (randomizer <= 9325) {

peptidesequence = 'Y';

} else {

peptidesequence = 'V';

}

protein();

}

}

function randomize\_fat() {

randomizer = math\_random\_int(1, 10000);

if (randomizer <= 10) {

randomize\_lipid\_3();

} else if (randomizer <= 3000) {

arachidonic\_acid();

} else if (randomizer <= 6000) {

alpha\_linoleic\_acid();

} else {

linoleic\_acid();

}

}

function randomize\_ribonucleotide() {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

ATP();

} else if (randomizer <= 50) {

cytosine\_mononucleotide();

} else if (randomizer <= 75) {

guanine\_mononucleotide();

} else {

uracil\_mononucleotide();

}

}

function randomize\_vitamin() {

randomizer = math\_random\_int(1, (15000 + (110 + (15000 + (15 + (90000 + (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2)))))))))))))) \* 1e+30);

if (randomizer <= 2 \* 1e+30) {

moco();

} else if (randomizer <= (1 + 2) \* 1e+30) {

cobalamin();

} else if (randomizer <= (700 + (1 + 2)) \* 1e+30) {

retinal();

} else if (randomizer <= (1200 + (700 + (1 + 2))) \* 1e+30) {

thiamine();

} else if (randomizer <= (1300 + (1200 + (700 + (1 + 2)))) \* 1e+30) {

riboflavin();

} else if (randomizer <= (16000 + (1300 + (1200 + (700 + (1 + 2))))) \* 1e+30) {

niacin();

} else if (randomizer <= (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2)))))) \* 1e+30) {

pantothenic\_acid();

} else if (randomizer <= (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2))))))) \* 1e+30) {

pyridoxine();

} else if (randomizer <= (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2)))))))) \* 1e+30) {

biotin();

} else if (randomizer <= (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2))))))))) \* 1e+30) {

folate();

} else if (randomizer <= (90000 + (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2)))))))))) \* 1e+30) {

ascorbic\_acid();

} else if (randomizer <= (15 + (90000 + (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2))))))))))) \* 1e+30) {

ergocalciferol();

} else if (randomizer <= (15000 + (15 + (90000 + (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2)))))))))))) \* 1e+30) {

tocopherol();

} else if (randomizer <= (110 + (15000 + (15 + (90000 + (400 + (30 + (1300 + (5000 + (16000 + (1300 + (1200 + (700 + (1 + 2))))))))))))) \* 1e+30) {

phylloquinone();

} else {

heme();

}

}

function ant\_genome2() {

genome = ant\_genome;

}

function aphid\_genome2() {

genome = aphid\_genome;

}

function bacillus\_genome2() {

genome = bacillus\_genome;

}

function bee\_genome2() {

genome = bee\_genome;

}

function conifer\_genome() {

genome = spruce\_genome;

}

function convert\_coordinates() {

if (relative == true) {

centerpoint\_x = centerpoint\_x + planet\_x;

centerpoint\_y = centerpoint\_y + planet\_y;

centerpoint\_z = centerpoint\_z + planet\_z;

}

}

function grass\_genome2() {

randomizer = math\_random\_int(1, 1000);

if (randomizer < 5) {

genome = plant\_genomes[math\_random\_int(1, plant\_genomes.length) - 1];

} else {

genome = grass\_genome;

}

}

function greenpatch\_genome() {

randomizer = math\_random\_int(1, 1000);

if (randomizer == 5) {

bacillus\_genome2();

} else {

spirulina\_genome2();

}

}

function human\_genome2() {

genome = human\_genome;

}

function lichen\_cyanobiont\_genome2() {

genome = lichen\_cyanobiont\_genome;

}

function lichen\_symbiont\_genome2() {

genome = lichen\_symbiont\_genome;

}

function nematode\_genome2() {

genome = nematode\_genome;

}

function pelagibacter\_genome2() {

genome = pelagibacter\_genome;

}

function randomize\_eukaryotic\_genome() {

genome = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

}

function randomize\_microbial\_genome() {

genome = prokaryotic\_genomes[math\_random\_int(1, prokaryotic\_genomes.length) - 1];

}

function rotifer\_genome2() {

genome = rotifer\_genome;

}

function set\_cell\_rotation() {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

set\_earth\_rotation();

}

function set\_organism\_rotation() {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

set\_earth\_rotation();

}

function shrub\_genome2() {

randomizer = math\_random\_int(1, 1000);

if (randomizer < 200) {

genome = plant\_genomes[math\_random\_int(1, plant\_genomes.length) - 1];

} else {

genome = shrub\_genome;

}

}

function spirulina\_genome2() {

genome = spirulina\_genome;

}

function tardigrade\_genome2() {

genome = tardigrade\_genome;

}

function tree\_genome2() {

randomizer = math\_random\_int(1, 1000);

if (randomizer == 5) {

genome = plant\_genomes[math\_random\_int(1, plant\_genomes.length) - 1];

} else {

genome = tree\_genome;

}

}

function villager\_female() {

ploidy = 2;

sizemax = 1.082887286e+35 - math\_random\_int(1, 1.23758547e+34);

human\_genome2();

human\_female();

}

function villager\_male() {

ploidy = 2;

sizemax = 1.082887286e+35 - math\_random\_int(1, 1.23758547e+34);

human\_genome2();

human\_male();

}

function flora() {

material\_temperature\_\_kelvins = 309.5;

rho = 3.942347699e+41;

relative = true;

var repeat\_end271 = 5000000000000000000 / 33;

for (var count271 = 0; count271 < repeat\_end271; count271++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

grass();

}

}

}

for (var count272 = 0; count272 < 18000000000000; count272++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

shrub\_genome2();

tree();

}

}

}

for (var count273 = 0; count273 < 18000000000000; count273++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

if (cursor\_phi > 0.1333 \* Math.PI) {

if (cursor\_phi < 0.2222222 \* Math.PI) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

conifer\_genome();

tree();

}

}

}

}

}

for (var count274 = 0; count274 < 18000000000000; count274++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

tree\_genome2();

tree();

}

}

}

fauna();

plankton();

algae();

village();

}

function fauna() {

material\_temperature\_\_kelvins = 309.5;

rho = 3.942347699e+41;

for (var count275 = 0; count275 < 50000000; count275++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

if (cursor\_phi < 0.75 \* Math.PI) {

if (cursor\_phi > 0.2222222 \* Math.PI) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

beehive();

}

}

}

}

}

for (var count276 = 0; count276 < 50000000; count276++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

if (altitude > 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

if (cursor\_phi < 0.75 \* Math.PI) {

if (cursor\_phi > 0.2222222 \* Math.PI) {

rho = altitude + 6.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

anthill();

}

}

}

}

}

}

function query\_plant\_location() {

permit = true;

if (cursor\_phi \* (180 / Math.PI) - 90 > -21.4638506) {

if (cursor\_phi \* (180 / Math.PI) - 90 < -20.860685692) {

if (cursor\_eta \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (cursor\_eta \* (180 / Math.PI) - 180 < 89.12348421518035) {

permit = false;

}

}

}

}

}

function anthill() {

relative = true;

convert\_coordinates();

material\_temperature\_\_kelvins = 309.5;

ant\_genome2();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end277 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count277 = 0; count277 < repeat\_end277; count277++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count278 = 0; count278 < generation\_parameter\_3; count278++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count279 = 0; count279 < generation\_parameter\_3; count279++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count280 = 0; count280 < generation\_parameter\_3; count280++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 6.187927353e+33;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end281 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count281 = 0; count281 < repeat\_end281; count281++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count282 = 0; count282 < generation\_parameter\_3; count282++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count283 = 0; count283 < generation\_parameter\_3; count283++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count284 = 0; count284 < generation\_parameter\_3; count284++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 6.187927353e+33;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end285 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count285 = 0; count285 < repeat\_end285; count285++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count286 = 0; count286 < generation\_parameter\_3; count286++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count287 = 0; count287 < generation\_parameter\_3; count287++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count288 = 0; count288 < generation\_parameter\_3; count288++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 9.187927353e+34;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end289 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count289 = 0; count289 < repeat\_end289; count289++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+34) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 1.299464744e+34) {

if (string\_z < centerpoint\_z + 6.187927353e+34) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count290 = 0; count290 < generation\_parameter\_3; count290++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count291 = 0; count291 < generation\_parameter\_y; count291++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count292 = 0; count292 < generation\_parameter\_3; count292++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

centerpoint\_y = centerpoint\_y - 6.187927353e+33;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+32 / used\_density\_parameter);

for (var count293 = 0; count293 < generation\_parameter\_3; count293++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count294 = 0; count294 < generation\_parameter\_y; count294++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count295 = 0; count295 < generation\_parameter\_3; count295++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count296 = 0; count296 < generation\_parameter\_3; count296++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count297 = 0; count297 < generation\_parameter\_y; count297++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count298 = 0; count298 < generation\_parameter\_3; count298++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count299 = 0; count299 < generation\_parameter\_3; count299++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count300 = 0; count300 < generation\_parameter\_y; count300++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count301 = 0; count301 < generation\_parameter\_3; count301++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count302 = 0; count302 < generation\_parameter\_3; count302++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count303 = 0; count303 < generation\_parameter\_y; count303++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count304 = 0; count304 < generation\_parameter\_3; count304++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

centerpoint\_y = centerpoint\_y - 6.187927353e+32;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count305 = 0; count305 < generation\_parameter\_3; count305++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count306 = 0; count306 < generation\_parameter\_y; count306++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count307 = 0; count307 < generation\_parameter\_3; count307++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

sizemax = 4.331549147e+32;

ploidy = 2;

cursor\_eta = cursor\_eta + 0.00002157894 \* (Math.PI / 180);

rho = altitude + 7.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

relative = true;

convert\_coordinates();

relative = false;

for (var count308 = 0; count308 < 10; count308++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count309 = 0; count309 < 10; count309++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count310 = 0; count310 < 10; count310++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_female();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

ploidy = 1;

centerpoint\_y = centerpoint\_y + 6.187927353e+32 \* 20.1;

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 20.1;

for (var count311 = 0; count311 < 10; count311++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count312 = 0; count312 < 10; count312++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count313 = 0; count313 < 10; count313++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_male();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

ploidy = 2;

sizemax = 2.475170941e+32;

aphid\_genome2();

cursor\_eta = cursor\_eta + 0.00002157894 \* (Math.PI / 180);

rho = altitude + 7.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

relative = true;

convert\_coordinates();

relative = false;

for (var count314 = 0; count314 < 10; count314++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count315 = 0; count315 < 10; count315++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count316 = 0; count316 < 10; count316++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_male();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

centerpoint\_y = centerpoint\_y + 6.187927353e+32 \* 20.1;

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 20.1;

for (var count317 = 0; count317 < 10; count317++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count318 = 0; count318 < 10; count318++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count319 = 0; count319 < 10; count319++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_female();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

genome = buchnera\_genome;

centerpoint\_y = centerpoint\_y + 6.187927353e+32 \* 20.1;

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 20.1;

for (var count320 = 0; count320 < 10; count320++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count321 = 0; count321 < 10; count321++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count322 = 0; count322 < 10; count322++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_female();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

}

relative = true;

}

function beehive() {

relative = true;

convert\_coordinates();

material\_temperature\_\_kelvins = 309.5;

bee\_genome2();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end323 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count323 = 0; count323 < repeat\_end323; count323++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count324 = 0; count324 < generation\_parameter\_3; count324++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count325 = 0; count325 < generation\_parameter\_3; count325++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count326 = 0; count326 < generation\_parameter\_3; count326++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 6.187927353e+33;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end327 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count327 = 0; count327 < repeat\_end327; count327++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count328 = 0; count328 < generation\_parameter\_3; count328++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count329 = 0; count329 < generation\_parameter\_3; count329++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count330 = 0; count330 < generation\_parameter\_3; count330++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 6.187927353e+33;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end331 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count331 = 0; count331 < repeat\_end331; count331++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+33) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 6.187927353e+24;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count332 = 0; count332 < generation\_parameter\_3; count332++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count333 = 0; count333 < generation\_parameter\_3; count333++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count334 = 0; count334 < generation\_parameter\_3; count334++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

centerpoint\_z = centerpoint\_z + 9.187927353e+34;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end335 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count335 = 0; count335 < repeat\_end335; count335++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+34) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 1.299464744e+34) {

if (string\_z < centerpoint\_z + 6.187927353e+34) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count336 = 0; count336 < generation\_parameter\_3; count336++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count337 = 0; count337 < generation\_parameter\_y; count337++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count338 = 0; count338 < generation\_parameter\_3; count338++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

centerpoint\_y = centerpoint\_y - 6.187927353e+33;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+32 / used\_density\_parameter);

for (var count339 = 0; count339 < generation\_parameter\_3; count339++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count340 = 0; count340 < generation\_parameter\_y; count340++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count341 = 0; count341 < generation\_parameter\_3; count341++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count342 = 0; count342 < generation\_parameter\_3; count342++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count343 = 0; count343 < generation\_parameter\_y; count343++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count344 = 0; count344 < generation\_parameter\_3; count344++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count345 = 0; count345 < generation\_parameter\_3; count345++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count346 = 0; count346 < generation\_parameter\_y; count346++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count347 = 0; count347 < generation\_parameter\_3; count347++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count348 = 0; count348 < generation\_parameter\_3; count348++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count349 = 0; count349 < generation\_parameter\_y; count349++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count350 = 0; count350 < generation\_parameter\_3; count350++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

centerpoint\_y = centerpoint\_y - 6.187927353e+32;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count351 = 0; count351 < generation\_parameter\_3; count351++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count352 = 0; count352 < generation\_parameter\_y; count352++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count353 = 0; count353 < generation\_parameter\_3; count353++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

sizemax = 1.23758547e+33;

ploidy = 2;

cursor\_eta = cursor\_eta + 0.00002157894 \* (Math.PI / 180);

rho = altitude + 7.187927353e+34;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

relative = true;

convert\_coordinates();

relative = false;

for (var count354 = 0; count354 < 10; count354++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count355 = 0; count355 < 10; count355++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count356 = 0; count356 < 10; count356++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_female();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

ploidy = 1;

centerpoint\_y = centerpoint\_y + 6.187927353e+32 \* 20.1;

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 20.1;

for (var count357 = 0; count357 < 10; count357++) {

centerpoint\_y = centerpoint\_y - 6.187927353e+32 \* 2.01;

for (var count358 = 0; count358 < 10; count358++) {

centerpoint\_x = centerpoint\_x - 6.187927353e+32 \* 2.01;

for (var count359 = 0; count359 < 10; count359++) {

centerpoint\_z = centerpoint\_z + 6.187927353e+32 \* 2.01;

set\_earth\_rotation();

invertebrate\_male();

}

centerpoint\_z = centerpoint\_z - 6.187927353e+32 \* 20.1;

}

centerpoint\_x = centerpoint\_x + 6.187927353e+32 \* 20.1;

}

}

relative = true;

}

function algae() {

material\_temperature\_\_kelvins = 309.5;

rho = 3.942266637e+41;

var repeat\_end360 = 5000000000000000000 / 33;

for (var count360 = 0; count360 < repeat\_end360; count360++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

permit = false;

if (cursor\_phi \* (180 / Math.PI) - 90 > -21.4638506) {

if (cursor\_phi \* (180 / Math.PI) - 90 < -20.8718435868) {

if (cursor\_eta \* (180 / Math.PI) - 180 > 86.79444293224275) {

if (cursor\_eta \* (180 / Math.PI) - 180 < 89.12348421518035) {

if (cursor\_eta \* (180 / Math.PI) - 180 < 86.9497172158) {

permit = true;

} else if (cursor\_eta \* (180 / Math.PI) - 180 > 86.9657172158) {

permit = true;

}

}

}

}

}

if (permit == true) {

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

greenpatch();

}

}

}

function plankton() {

material\_temperature\_\_kelvins = 309.5;

rho = 3.942347699e+41;

for (var count361 = 0; count361 < 50000000000; count361++) {

cursor\_eta = (math\_random\_int(1000, 3.59999999999e+75) / 1e+73) \* (Math.PI / 180);

cursor\_phi = (math\_random\_int(1000, 1.7999e+75) / 1e+73) \* (Math.PI / 180);

lunar\_eta\_coordinate = cursor\_eta;

lunar\_phi\_coordinate = cursor\_phi;

obtain\_altitude();

query\_plankton\_location();

if (permit == true) {

if (altitude < 3.942328517e+41) {

query\_plant\_location();

if (permit == true) {

rho = 3.942328517e+41;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

convert\_coordinates();

phytoplankton();

}

}

}

}

}

function grass() {

ploidy = 2;

grass\_genome2();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end362 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count362 = 0; count362 < repeat\_end362; count362++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+31) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 3.093963676e+32) {

if (string\_z < centerpoint\_z + 6.187927353e+31) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 120);

for (var count363 = 0; count363 < generation\_parameter\_3; count363++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count364 = 0; count364 < generation\_parameter\_3; count364++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count365 = 0; count365 < generation\_parameter\_z; count365++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 60);

for (var count366 = 0; count366 < generation\_parameter\_3; count366++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count367 = 0; count367 < generation\_parameter\_3; count367++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count368 = 0; count368 < generation\_parameter\_z; count368++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1z = centerpoint\_z + 3.093963676e+31;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 120);

for (var count369 = 0; count369 < generation\_parameter\_3; count369++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count370 = 0; count370 < generation\_parameter\_3; count370++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count371 = 0; count371 < generation\_parameter\_z; count371++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1z = centerpoint\_z + 3.093963676e+31;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 60);

for (var count372 = 0; count372 < generation\_parameter\_3; count372++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count373 = 0; count373 < generation\_parameter\_3; count373++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count374 = 0; count374 < generation\_parameter\_z; count374++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

root\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+31;

spawnerz = centerpoint\_z;

}

var repeat\_end375 = Math.round(6.187927353e+31 / 6.187927353e+29 - 2);

for (var count375 = 0; count375 < repeat\_end375; count375++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end376 = Math.round(1.856378206e+32 / 6.187927353e+29 - 2);

for (var count376 = 0; count376 < repeat\_end376; count376++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end377 = Math.round(6.187927353e+31 / 6.187927353e+29 - 2);

for (var count377 = 0; count377 < repeat\_end377; count377++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(1.856378206e+31 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+32;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 120);

for (var count378 = 0; count378 < generation\_parameter\_3; count378++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count379 = 0; count379 < generation\_parameter\_3; count379++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count380 = 0; count380 < generation\_parameter\_z; count380++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+32;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 60);

for (var count381 = 0; count381 < generation\_parameter\_3; count381++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count382 = 0; count382 < generation\_parameter\_3; count382++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count383 = 0; count383 < generation\_parameter\_z; count383++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+32;

cursor1z = centerpoint\_z + 3.093963676e+31;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 120);

for (var count384 = 0; count384 < generation\_parameter\_3; count384++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count385 = 0; count385 < generation\_parameter\_3; count385++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count386 = 0; count386 < generation\_parameter\_z; count386++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+32;

cursor1z = centerpoint\_z + 3.093963676e+31;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+31 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+31 / used\_density\_parameter - 60);

for (var count387 = 0; count387 < generation\_parameter\_3; count387++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count388 = 0; count388 < generation\_parameter\_3; count388++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count389 = 0; count389 < generation\_parameter\_z; count389++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

shoot\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function tree() {

ploidy = 2;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end390 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count390 = 0; count390 < repeat\_end390; count390++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+32) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 3.093963676e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+32) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 120);

for (var count391 = 0; count391 < generation\_parameter\_3; count391++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count392 = 0; count392 < generation\_parameter\_3; count392++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count393 = 0; count393 < generation\_parameter\_z; count393++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 60);

for (var count394 = 0; count394 < generation\_parameter\_3; count394++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count395 = 0; count395 < generation\_parameter\_3; count395++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count396 = 0; count396 < generation\_parameter\_z; count396++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1z = centerpoint\_z + 3.093963676e+32;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 120);

for (var count397 = 0; count397 < generation\_parameter\_3; count397++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count398 = 0; count398 < generation\_parameter\_3; count398++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count399 = 0; count399 < generation\_parameter\_z; count399++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1z = centerpoint\_z + 3.093963676e+32;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 60);

for (var count400 = 0; count400 < generation\_parameter\_3; count400++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count401 = 0; count401 < generation\_parameter\_3; count401++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count402 = 0; count402 < generation\_parameter\_z; count402++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

root\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+32;

spawnerz = centerpoint\_z;

}

var repeat\_end403 = Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

for (var count403 = 0; count403 < repeat\_end403; count403++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end404 = Math.round(1.856378206e+33 / 6.187927353e+29 - 2);

for (var count404 = 0; count404 < repeat\_end404; count404++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end405 = Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

for (var count405 = 0; count405 < repeat\_end405; count405++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(1.856378206e+32 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+33 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+33;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 120);

for (var count406 = 0; count406 < generation\_parameter\_3; count406++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count407 = 0; count407 < generation\_parameter\_3; count407++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count408 = 0; count408 < generation\_parameter\_z; count408++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+33;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 60);

for (var count409 = 0; count409 < generation\_parameter\_3; count409++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count410 = 0; count410 < generation\_parameter\_3; count410++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count411 = 0; count411 < generation\_parameter\_z; count411++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+33;

cursor1z = centerpoint\_z + 3.093963676e+32;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 120);

for (var count412 = 0; count412 < generation\_parameter\_3; count412++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count413 = 0; count413 < generation\_parameter\_3; count413++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count414 = 0; count414 < generation\_parameter\_z; count414++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 2.475170941e+33;

cursor1z = centerpoint\_z + 3.093963676e+32;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(6.187927353e+32 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+32 / used\_density\_parameter - 60);

for (var count415 = 0; count415 < generation\_parameter\_3; count415++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count416 = 0; count416 < generation\_parameter\_3; count416++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count417 = 0; count417 < generation\_parameter\_z; count417++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

shoot\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function greenpatch() {

ploidy = 1;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end418 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count418 = 0; count418 < repeat\_end418; count418++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+34) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+33) {

if (string\_z < centerpoint\_z + 6.187927353e+34) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

var repeat\_end419 = Math.round(6.187927353e+34 / 6.187927353e+29 - 2);

for (var count419 = 0; count419 < repeat\_end419; count419++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end420 = Math.round(6.187927353e+33 / 6.187927353e+29 - 2);

for (var count420 = 0; count420 < repeat\_end420; count420++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end421 = Math.round(6.187927353e+34 / 6.187927353e+29 - 2);

for (var count421 = 0; count421 < repeat\_end421; count421++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

greenpatch\_genome();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+34 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+33 / 6.187927353e+29 - 2);

}

}

}

}

function phytoplankton() {

ploidy = 1;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end422 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count422 = 0; count422 < repeat\_end422; count422++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+33) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.187927353e+32) {

if (string\_z < centerpoint\_z + 6.187927353e+32) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

var repeat\_end423 = Math.round(6.187927353e+33 / 6.187927353e+29 - 2);

for (var count423 = 0; count423 < repeat\_end423; count423++) {

cursor1x = cursor1x - 6.187927353e+29;

var repeat\_end424 = Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

for (var count424 = 0; count424 < repeat\_end424; count424++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end425 = Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

for (var count425 = 0; count425 < repeat\_end425; count425++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

greenpatch\_genome();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+32 / 6.187927353e+29 - 2);

}

}

}

}

function query\_plankton\_location() {

permit = true;

if (altitude > 3.942328517e+41 - Math.abs(lunar\_phi\_coordinate \* (180 / Math.PI) - 90) \* (1.23758547e+37 / 90)) {

permit = false;

}

}

function village() {

material\_temperature\_\_kelvins = 273.18;

rho = 3.942347699e+41;

cursor\_phi = (90 - 20.8714225341) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9577172158) \* (Math.PI / 180);

cursor1x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

cursor1y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

cursor1z = rho \* Math.cos(cursor\_phi);

for (var count426 = 0; count426 < 20; count426++) {

step\_eta();

house\_1();

}

reset\_eta();

step\_phi();

for (var count427 = 0; count427 < 20; count427++) {

step\_eta();

house\_1();

}

reset\_eta();

step\_phi();

for (var count428 = 0; count428 < 20; count428++) {

step\_eta();

house\_1();

}

reset\_eta();

step\_phi();

for (var count429 = 0; count429 < 20; count429++) {

step\_eta();

house\_1();

}

reset\_eta();

step\_phi();

for (var count430 = 0; count430 < 20; count430++) {

step\_eta();

house\_1();

}

reset\_eta();

step\_phi();

for (var count431 = 0; count431 < 20; count431++) {

step\_eta();

house\_2();

}

reset\_eta();

step\_phi();

for (var count432 = 0; count432 < 20; count432++) {

step\_eta();

house\_2();

}

reset\_eta();

step\_phi();

for (var count433 = 0; count433 < 20; count433++) {

step\_eta();

house\_3();

}

reset\_eta();

step\_phi();

for (var count434 = 0; count434 < 20; count434++) {

step\_eta();

house\_4();

}

reset\_eta();

step\_phi();

for (var count435 = 0; count435 < 20; count435++) {

step\_eta();

house\_5();

}

reset\_eta();

step\_phi();

for (var count436 = 0; count436 < 20; count436++) {

step\_eta();

house\_6();

}

reset\_eta();

step\_phi();

for (var count437 = 0; count437 < 20; count437++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count438 = 0; count438 < 20; count438++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count439 = 0; count439 < 20; count439++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count440 = 0; count440 < 20; count440++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count441 = 0; count441 < 20; count441++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count442 = 0; count442 < 20; count442++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count443 = 0; count443 < 20; count443++) {

step\_eta();

house\_8();

}

reset\_eta();

step\_phi();

for (var count444 = 0; count444 < 20; count444++) {

step\_eta();

house\_9();

}

reset\_eta();

step\_phi();

for (var count445 = 0; count445 < 20; count445++) {

step\_eta();

house\_10();

}

reset\_eta();

step\_phi();

for (var count446 = 0; count446 < 20; count446++) {

step\_eta();

house\_10();

}

reset\_eta();

step\_phi();

for (var count447 = 0; count447 < 20; count447++) {

step\_eta();

house\_10();

}

reset\_eta();

step\_phi();

for (var count448 = 0; count448 < 20; count448++) {

step\_eta();

house\_11();

}

reset\_eta();

step\_phi();

for (var count449 = 0; count449 < 20; count449++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count450 = 0; count450 < 20; count450++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count451 = 0; count451 < 20; count451++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count452 = 0; count452 < 20; count452++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count453 = 0; count453 < 20; count453++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

unused\_variable = 5;

for (var count454 = 0; count454 < 20; count454++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count455 = 0; count455 < 20; count455++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count456 = 0; count456 < 20; count456++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count457 = 0; count457 < 20; count457++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count458 = 0; count458 < 20; count458++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

unused\_variable = 5;

for (var count459 = 0; count459 < 20; count459++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count460 = 0; count460 < 20; count460++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count461 = 0; count461 < 20; count461++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count462 = 0; count462 < 20; count462++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

for (var count463 = 0; count463 < 20; count463++) {

step\_eta();

house\_7();

}

reset\_eta();

step\_phi();

relative = true;

inhabitants();

additions();

terrestrial();

freshwater();

marine();

relative = false;

}

function reset\_eta() {

cursor\_eta = cursor\_eta + (0.00021052631 \* 20) \* (Math.PI / 180);

}

function step\_eta() {

cursor\_eta = cursor\_eta - 0.00021052631 \* (Math.PI / 180);

locate();

}

function step\_phi() {

cursor\_phi = cursor\_phi + 0.00021052631 \* (Math.PI / 180);

}

function locate() {

relative\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

relative\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

relative\_z = rho \* Math.cos(cursor\_phi);

cursor1x = planet\_x + relative\_x;

cursor1y = planet\_y + relative\_y;

cursor1z = planet\_z + relative\_z;

set\_earth\_rotation();

}

function house\_1() {

unused\_variable = 5;

create\_building();

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 + 3.093963676e+32);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+32);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+32);

for (var count507 = 0; count507 < 7; count507++) {

for (var count508 = 0; count508 < 4; count508++) {

bed();

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 2.75;

}

centerpoint\_z = centerpoint\_z - ((3.093963676e+34 \* 2) \* 2.75) \* 4;

centerpoint\_x = centerpoint\_x + (3.093963676e+34 \* 2) \* 1.25;

}

}

}

function create\_building() {

if (unused\_variable == 5) {

centerpoint\_x = cursor1x;

centerpoint\_y = cursor1y;

centerpoint\_z = cursor1z;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end464 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count464 = 0; count464 < repeat\_end464; count464++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 8.663098295e+35) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 8.663098295e+35) {

if (string\_z < centerpoint\_z + 8.663098295e+35) {

if (string\_z > centerpoint\_z) {

if (Math.abs(centerpoint\_y - string\_y) < 3.093963676e+34) {

if (Math.abs(centerpoint\_y - string\_y) < Math.abs(centerpoint\_z - string\_z) / 4) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

} else if (Math.abs(string\_z - centerpoint\_z) > 8.044305559e+35) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(centerpoint\_y - string\_y) < 4.486247331e+35) {

if (Math.abs(centerpoint\_x - string\_x) < 3.093963676e+34 \* 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(centerpoint\_x - string\_x) > 8.044305559e+35) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

} else if (Math.abs(centerpoint\_y - string\_y) > 4.486247331e+35) {

if (Math.abs(string\_z - centerpoint\_z) > 3.093963676e+34 \* 2) {

if (Math.abs(centerpoint\_y - string\_y) < ((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - string\_x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) {

if (Math.abs(centerpoint\_y - string\_y) > (((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - string\_x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) - 3.093963676e+34 \* 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

for (var count465 = 0; count465 < generation\_parameter\_3; count465++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count466 = 0; count466 < generation\_parameter\_3; count466++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count467 = 0; count467 < generation\_parameter\_3; count467++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 3.093963676e+34) {

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs(centerpoint\_z - cursor1z) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(cursor1z - centerpoint\_z) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_y - cursor1y) < 4.486247331e+35) {

if (Math.abs(centerpoint\_x - cursor1x) < 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(centerpoint\_y - cursor1y) > 4.486247331e+35) {

if (Math.abs(cursor1z - centerpoint\_z) > 3.093963676e+34 \* 2) {

if (Math.abs(centerpoint\_y - cursor1y) < ((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) {

if (Math.abs(centerpoint\_y - cursor1y) > (((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) - 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count468 = 0; count468 < generation\_parameter\_3; count468++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count469 = 0; count469 < generation\_parameter\_3; count469++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count470 = 0; count470 < generation\_parameter\_3; count470++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 3.093963676e+34) {

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs(centerpoint\_z - cursor1z) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(cursor1z - centerpoint\_z) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_y - cursor1y) < 4.486247331e+35) {

if (Math.abs(centerpoint\_x - cursor1x) < 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(centerpoint\_y - cursor1y) > 4.486247331e+35) {

if (Math.abs(cursor1z - centerpoint\_z) > 3.093963676e+34 \* 2) {

if (Math.abs(centerpoint\_y - cursor1y) < ((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) {

if (Math.abs(centerpoint\_y - cursor1y) > (((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) - 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count471 = 0; count471 < generation\_parameter\_3; count471++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count472 = 0; count472 < generation\_parameter\_3; count472++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count473 = 0; count473 < generation\_parameter\_3; count473++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 3.093963676e+34) {

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs(centerpoint\_z - cursor1z) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(cursor1z - centerpoint\_z) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_y - cursor1y) < 4.486247331e+35) {

if (Math.abs(centerpoint\_x - cursor1x) < 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(centerpoint\_y - cursor1y) > 4.486247331e+35) {

if (Math.abs(cursor1z - centerpoint\_z) > 3.093963676e+34 \* 2) {

if (Math.abs(centerpoint\_y - cursor1y) < ((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) {

if (Math.abs(centerpoint\_y - cursor1y) > (((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) - 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count474 = 0; count474 < generation\_parameter\_3; count474++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count475 = 0; count475 < generation\_parameter\_3; count475++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count476 = 0; count476 < generation\_parameter\_3; count476++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 3.093963676e+34) {

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs(centerpoint\_z - cursor1z) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(cursor1z - centerpoint\_z) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_y - cursor1y) < 4.486247331e+35) {

if (Math.abs(centerpoint\_x - cursor1x) < 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 8.044305559e+35) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (Math.abs(centerpoint\_y - cursor1y) > 4.486247331e+35) {

if (Math.abs(cursor1z - centerpoint\_z) > 3.093963676e+34 \* 2) {

if (Math.abs(centerpoint\_y - cursor1y) < ((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) {

if (Math.abs(centerpoint\_y - cursor1y) > (((8.663098295e+35 / 2 - Math.abs(Math.abs(centerpoint\_x - cursor1x) - 8.663098295e+35 / 2)) / (8.663098295e+35 / 2)) \* 4.176850963e+35 + 4.486247331e+35) - 3.093963676e+34 \* 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function inhabitants() {

material\_temperature\_\_kelvins = 309.5;

rho = 3.942347699e+41;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9629803737) \* (Math.PI / 180);

cursor\_phi = cursor\_phi + 0.000071052631 \* (Math.PI / 180);

for (var count477 = 0; count477 < 20; count477++) {

cursor\_phi = cursor\_phi + 0.00007368421 \* (Math.PI / 180);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

villager\_male();

}

material\_temperature\_\_kelvins = 309.5;

rho = 3.942347699e+41;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9629803737) \* (Math.PI / 180);

cursor\_phi = cursor\_phi + 0.000071052631 \* (Math.PI / 180);

cursor\_eta = cursor\_eta - 0.000071052631 \* (Math.PI / 180);

for (var count478 = 0; count478 < 20; count478++) {

cursor\_phi = cursor\_phi + 0.00007368421 \* (Math.PI / 180);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

villager\_female();

}

}

function hollow\_ingot() {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end479 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count479 = 0; count479 < repeat\_end479; count479++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187865474e+33) {

if (string\_z < starter\_z + 6.187865474e+33) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

function carve\_container() {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end480 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count480 = 0; count480 < repeat\_end480; count480++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x - 6.187865474e+32) {

if (string\_x > (starter\_x - 6.187865474e+33) + 6.187865474e+32) {

if (string\_y < starter\_y - 6.187865474e+32) {

if (string\_y > (starter\_y - 6.187865474e+33) + 6.187865474e+32) {

if (string\_z < (starter\_z + 6.187865474e+33) - 6.187865474e+32) {

if (string\_z > starter\_z + 6.187865474e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

function container() {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end481 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count481 = 0; count481 < repeat\_end481; count481++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187865474e+33) {

if (string\_z < starter\_z + 6.187865474e+33) {

if (string\_z > starter\_z) {

if (string\_y > starter\_y - 5.569072739e+33) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

} else if (string\_y < starter\_y - 5.569134618e+33) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count482 = 0; count482 < generation\_parameter\_3; count482++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count483 = 0; count483 < generation\_parameter\_3; count483++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count484 = 0; count484 < generation\_parameter\_3; count484++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1y > starter\_y - 5.569072739e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 5.569134618e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count485 = 0; count485 < generation\_parameter\_3; count485++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count486 = 0; count486 < generation\_parameter\_3; count486++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count487 = 0; count487 < generation\_parameter\_3; count487++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1y > starter\_y - 5.569072739e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 5.569134618e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count488 = 0; count488 < generation\_parameter\_3; count488++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count489 = 0; count489 < generation\_parameter\_3; count489++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count490 = 0; count490 < generation\_parameter\_3; count490++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1y > starter\_y - 5.569072739e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 5.569134618e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count491 = 0; count491 < generation\_parameter\_3; count491++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count492 = 0; count492 < generation\_parameter\_3; count492++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count493 = 0; count493 < generation\_parameter\_3; count493++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1y > starter\_y - 5.569072739e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 5.569134618e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

function package2() {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end494 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count494 = 0; count494 < repeat\_end494; count494++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.373565174e+34) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 6.373565174e+34) {

if (string\_z < centerpoint\_z + 6.373565174e+34) {

if (string\_z > centerpoint\_z) {

if (Math.abs(centerpoint\_y - string\_y) < 6.187927353e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(cursor1z - string\_z) < 6.187927353e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(centerpoint\_x - string\_x) < 6.187927353e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(centerpoint\_x - string\_x) > 6.3116859e+34) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.373565174e+34 / used\_density\_parameter);

for (var count495 = 0; count495 < generation\_parameter\_3; count495++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count496 = 0; count496 < generation\_parameter\_3; count496++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count497 = 0; count497 < generation\_parameter\_3; count497++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(cursor1z - centerpoint\_z) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 6.3116859e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.373565174e+34 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count498 = 0; count498 < generation\_parameter\_3; count498++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count499 = 0; count499 < generation\_parameter\_3; count499++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count500 = 0; count500 < generation\_parameter\_3; count500++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(cursor1z - centerpoint\_z) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 6.3116859e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.373565174e+34 / used\_density\_parameter);

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count501 = 0; count501 < generation\_parameter\_3; count501++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count502 = 0; count502 < generation\_parameter\_3; count502++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count503 = 0; count503 < generation\_parameter\_3; count503++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(cursor1z - centerpoint\_z) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 6.3116859e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(6.373565174e+34 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count504 = 0; count504 < generation\_parameter\_3; count504++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count505 = 0; count505 < generation\_parameter\_3; count505++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count506 = 0; count506 < generation\_parameter\_3; count506++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(cursor1z - centerpoint\_z) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) < 6.187927353e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(centerpoint\_x - cursor1x) > 6.3116859e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

function bed() {

unused\_variable = 5;

set\_earth\_rotation();

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count509 = 0; count509 < 500; count509++) {

for (var count510 = 0; count510 < 1000; count510++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count511 = 0; count511 < string\_solar\_distance\_\_length\_parameter; count511++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 2e+31) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 2e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34 \* 2) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(2e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(2e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+34 \* 2) / density\_parameter\_z);

for (var count512 = 0; count512 < generation\_parameter\_x; count512++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count513 = 0; count513 < generation\_parameter\_y; count513++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count514 = 0; count514 < generation\_parameter\_z; count514++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31;

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

}

starter\_y = starter\_y - 6.187865474e+31;

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = starter\_x - (3.093963676e+30 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count515 = 0; count515 < 2000; count515++) {

for (var count516 = 0; count516 < 490; count516++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count517 = 0; count517 < string\_solar\_distance\_\_length\_parameter; count517++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34 \* 1) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 1e+31) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round((6.187865474e+34 \* 1) / density\_parameter\_x);

generation\_parameter\_y = Math.round(1e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(1e+31 / density\_parameter\_z);

for (var count518 = 0; count518 < generation\_parameter\_x; count518++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count519 = 0; count519 < generation\_parameter\_y; count519++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count520 = 0; count520 < generation\_parameter\_z; count520++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31;

}

starter\_x = starter\_x - 6.187865474e+31;

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = starter\_x - (3.093963676e+30 + 3.093963676e+30);

starter\_z = starter\_z + (3.093963676e+30 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count521 = 0; count521 < 2000; count521++) {

for (var count522 = 0; count522 < 1000; count522++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count523 = 0; count523 < string\_solar\_distance\_\_length\_parameter; count523++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_y - 6e+30) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187865474e+34 \* 0.48) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6e+30) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6e+30 / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+34 \* 0.48) / density\_parameter\_y);

generation\_parameter\_z = Math.round(6e+30 / density\_parameter\_z);

for (var count524 = 0; count524 < generation\_parameter\_x; count524++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count525 = 0; count525 < generation\_parameter\_y; count525++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count526 = 0; count526 < generation\_parameter\_z; count526++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31;

}

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

starter\_z = starter\_z + 6.187865474e+31;

}

}

}

}

}

function additions() {

material\_temperature\_\_kelvins = 273.18;

rho = 3.942347699e+41;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9629803737) \* (Math.PI / 180);

cursor\_eta = cursor\_phi - 0.000031052631 \* (Math.PI / 180);

var repeat\_end527 = Math.floor((21.4638506 - 20.8718435868) / 0.00021052631) - 3;

for (var count527 = 0; count527 < repeat\_end527; count527++) {

cursor\_phi = cursor\_phi - 0.00021052631 \* (Math.PI / 180);

centerpoint\_x = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.cos(cursor\_eta / 180 \* Math.PI));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta / 180 \* Math.PI));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

convert\_coordinates();

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end528 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count528 = 0; count528 < repeat\_end528; count528++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+35) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 3.093963676e+34) {

if (string\_z < centerpoint\_z + 6.187927353e+35) {

if (string\_z > centerpoint\_z) {

if (Math.abs(centerpoint\_y - string\_y) < Math.abs((centerpoint\_x - 6.187927353e+35) - string\_x) / 2) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(3.093963676e+34 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+35 / used\_density\_parameter);

for (var count529 = 0; count529 < generation\_parameter\_x; count529++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count530 = 0; count530 < generation\_parameter\_y; count530++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count531 = 0; count531 < generation\_parameter\_z; count531++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs((centerpoint\_x - 6.187927353e+35) - cursor1x) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (var count532 = 0; count532 < generation\_parameter\_3; count532++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count533 = 0; count533 < generation\_parameter\_3; count533++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count534 = 0; count534 < generation\_parameter\_3; count534++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs((centerpoint\_x - 6.187927353e+35) - cursor1x) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count535 = 0; count535 < generation\_parameter\_3; count535++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count536 = 0; count536 < generation\_parameter\_3; count536++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count537 = 0; count537 < generation\_parameter\_3; count537++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs((centerpoint\_x - 6.187927353e+35) - cursor1x) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_3 = Math.round(8.663098295e+35 / used\_density\_parameter);

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

for (var count538 = 0; count538 < generation\_parameter\_3; count538++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count539 = 0; count539 < generation\_parameter\_3; count539++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count540 = 0; count540 < generation\_parameter\_3; count540++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(centerpoint\_y - cursor1y) < Math.abs((centerpoint\_x - 6.187927353e+35) - cursor1x) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_x = centerpoint\_x;

starter\_y = centerpoint\_y - 3.093963676e+34;

starter\_z = centerpoint\_z;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end541 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count541 = 0; count541 < repeat\_end541; count541++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 5.569134618e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 2.475170941e+35) {

if (string\_z < starter\_z + 6.187927353e+34) {

if (string\_z > starter\_z) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(5.569134618e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(2.475170941e+35 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count542 = 0; count542 < generation\_parameter\_x; count542++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count543 = 0; count543 < generation\_parameter\_y; count543++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count544 = 0; count544 < generation\_parameter\_z; count544++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_z = centerpoint\_z + 5.569134618e+35;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end545 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count545 = 0; count545 < repeat\_end545; count545++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 5.569134618e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 2.475170941e+35) {

if (string\_z < starter\_z + 6.187927353e+34) {

if (string\_z > starter\_z) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(5.569134618e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(2.475170941e+35 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count546 = 0; count546 < generation\_parameter\_x; count546++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count547 = 0; count547 < generation\_parameter\_y; count547++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count548 = 0; count548 < generation\_parameter\_z; count548++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_z = centerpoint\_z + 6.187927353e+34;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end549 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count549 = 0; count549 < repeat\_end549; count549++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.856378206e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.856378206e+35) {

if (string\_z < starter\_z + 4.950341882e+35) {

if (string\_z > starter\_z) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.856378206e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.856378206e+35 / used\_density\_parameter);

generation\_parameter\_z = Math.round(4.950341882e+35 / used\_density\_parameter);

for (var count550 = 0; count550 < generation\_parameter\_x; count550++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count551 = 0; count551 < generation\_parameter\_y; count551++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count552 = 0; count552 < generation\_parameter\_z; count552++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_x = centerpoint\_x - 1.856378206e+35;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end553 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count553 = 0; count553 < repeat\_end553; count553++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.856378206e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.856378206e+35) {

if (string\_z < starter\_z + 4.950341882e+35) {

if (string\_z > starter\_z) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.856378206e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.856378206e+35 / used\_density\_parameter);

generation\_parameter\_z = Math.round(4.950341882e+35 / used\_density\_parameter);

for (var count554 = 0; count554 < generation\_parameter\_x; count554++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count555 = 0; count555 < generation\_parameter\_y; count555++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count556 = 0; count556 < generation\_parameter\_z; count556++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_x = starter\_x - 6.187927353e+34;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end557 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count557 = 0; count557 < repeat\_end557; count557++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187927353e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187927353e+34) {

if (string\_z < starter\_z + 4.950341882e+35) {

if (string\_z > starter\_z) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_z = Math.round(4.950341882e+35 / used\_density\_parameter);

for (var count558 = 0; count558 < generation\_parameter\_x; count558++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count559 = 0; count559 < generation\_parameter\_y; count559++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count560 = 0; count560 < generation\_parameter\_z; count560++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_x = starter\_x - 6.187927353e+34;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end561 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count561 = 0; count561 < repeat\_end561; count561++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.23758547e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187927353e+34) {

if (string\_z < starter\_z + 2.475170941e+35) {

if (string\_z > starter\_z) {

if (Math.abs(centerpoint\_y - string\_y) < Math.abs((centerpoint\_x - 1.23758547e+35) - string\_x) / 2) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.23758547e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_z = Math.round(2.475170941e+35 / used\_density\_parameter);

for (var count562 = 0; count562 < generation\_parameter\_x; count562++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count563 = 0; count563 < generation\_parameter\_y; count563++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count564 = 0; count564 < generation\_parameter\_z; count564++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_y - 1.23758547e+35) - cursor1x) / 2) {

set\_earth\_rotation();

randomize\_nutrient\_1();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

starter\_z = starter\_z + 2.475170941e+35;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end565 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count565 = 0; count565 < repeat\_end565; count565++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.23758547e+35) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187927353e+34) {

if (string\_z < starter\_z + 2.475170941e+35) {

if (string\_z > starter\_z) {

if (Math.abs(centerpoint\_y - string\_y) < Math.abs((centerpoint\_x - 1.23758547e+35) - string\_x) / 2) {

list.splice((string\_solar\_\_distance\_parameter + 1) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.23758547e+35 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_z = Math.round(2.475170941e+35 / used\_density\_parameter);

for (var count566 = 0; count566 < generation\_parameter\_x; count566++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count567 = 0; count567 < generation\_parameter\_y; count567++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count568 = 0; count568 < generation\_parameter\_z; count568++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_y - 1.23758547e+35) - cursor1x) / 2) {

set\_earth\_rotation();

randomize\_nutrient\_2();

}

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function sort\_ingot() {

iron\_metallic\_();

}

function house\_2() {

unused\_variable = 5;

create\_building();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* 6;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count569 = 0; count569 < 10; count569++) {

for (var count570 = 0; count570 < 10; count570++) {

for (var count571 = 0; count571 < 10; count571++) {

hollow\_ingot();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 5;

used\_density\_parameter = 1.773893134e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.187865474e+33 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187865474e+33 / used\_density\_parameter);

for (var count572 = 0; count572 < generation\_parameter\_x; count572++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count573 = 0; count573 < generation\_parameter\_y; count573++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count574 = 0; count574 < generation\_parameter\_z; count574++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

sort\_ingot();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1y + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count575 = 0; count575 < generation\_parameter\_x; count575++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count576 = 0; count576 < generation\_parameter\_y; count576++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count577 = 0; count577 < generation\_parameter\_z; count577++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

sort\_ingot();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

allotrope\_bars();

}

function allotrope\_bars() {

unused\_variable = 5;

set\_earth\_rotation();

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 295.18;

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count578 = 0; count578 < 10; count578++) {

for (var count579 = 0; count579 < 10; count579++) {

for (var count580 = 0; count580 < 10; count580++) {

hollow\_ingot();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 12;

density\_parameter\_x = 1.521920732e+25;

density\_parameter\_y = 4.150861668e+25;

density\_parameter\_z = 2.636057052e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count581 = 0; count581 < generation\_parameter\_x; count581++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count582 = 0; count582 < generation\_parameter\_y; count582++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count583 = 0; count583 < generation\_parameter\_z; count583++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 9.964110021e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 12;

for (var count584 = 0; count584 < generation\_parameter\_x; count584++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count585 = 0; count585 < generation\_parameter\_y; count585++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count586 = 0; count586 < generation\_parameter\_z; count586++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 1.177253179e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 13;

for (var count587 = 0; count587 < generation\_parameter\_x; count587++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count588 = 0; count588 < generation\_parameter\_y; count588++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count589 = 0; count589 < generation\_parameter\_z; count589++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 1.757371368e+25;

sorter = 13;

for (var count590 = 0; count590 < generation\_parameter\_x; count590++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count591 = 0; count591 < generation\_parameter\_y; count591++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count592 = 0; count592 < generation\_parameter\_z; count592++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 12;

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

for (var count593 = 0; count593 < generation\_parameter\_x; count593++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count594 = 0; count594 < generation\_parameter\_y; count594++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count595 = 0; count595 < generation\_parameter\_z; count595++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 9.964110021e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 12;

for (var count596 = 0; count596 < generation\_parameter\_x; count596++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count597 = 0; count597 < generation\_parameter\_y; count597++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count598 = 0; count598 < generation\_parameter\_z; count598++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 1.177253179e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 13;

for (var count599 = 0; count599 < generation\_parameter\_x; count599++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count600 = 0; count600 < generation\_parameter\_y; count600++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count601 = 0; count601 < generation\_parameter\_z; count601++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 1.757371368e+25;

sorter = 13;

for (var count602 = 0; count602 < generation\_parameter\_x; count602++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count603 = 0; count603 < generation\_parameter\_y; count603++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count604 = 0; count604 < generation\_parameter\_z; count604++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count605 = 0; count605 < 10; count605++) {

for (var count606 = 0; count606 < 10; count606++) {

for (var count607 = 0; count607 < 10; count607++) {

hollow\_ingot();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count608 = 0; count608 < generation\_parameter\_x; count608++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count609 = 0; count609 < generation\_parameter\_y; count609++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count610 = 0; count610 < generation\_parameter\_z; count610++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count611 = 0; count611 < generation\_parameter\_x; count611++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count612 = 0; count612 < generation\_parameter\_y; count612++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count613 = 0; count613 < generation\_parameter\_z; count613++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count614 = 0; count614 < generation\_parameter\_x; count614++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count615 = 0; count615 < generation\_parameter\_y; count615++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count616 = 0; count616 < generation\_parameter\_z; count616++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count617 = 0; count617 < generation\_parameter\_x; count617++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count618 = 0; count618 < generation\_parameter\_y; count618++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count619 = 0; count619 < generation\_parameter\_z; count619++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count620 = 0; count620 < generation\_parameter\_x; count620++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count621 = 0; count621 < generation\_parameter\_y; count621++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count622 = 0; count622 < generation\_parameter\_z; count622++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count623 = 0; count623 < generation\_parameter\_x; count623++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count624 = 0; count624 < generation\_parameter\_y; count624++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count625 = 0; count625 < generation\_parameter\_z; count625++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count626 = 0; count626 < generation\_parameter\_x; count626++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count627 = 0; count627 < generation\_parameter\_y; count627++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count628 = 0; count628 < generation\_parameter\_z; count628++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count629 = 0; count629 < generation\_parameter\_x; count629++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count630 = 0; count630 < generation\_parameter\_y; count630++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count631 = 0; count631 < generation\_parameter\_z; count631++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function house\_3() {

unused\_variable = 5;

set\_earth\_rotation();

create\_building();

}

function house\_6() {

unused\_variable = 5;

set\_earth\_rotation();

create\_building();

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count632 = 0; count632 < 1000; count632++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count633 = 0; count633 < string\_solar\_distance\_\_length\_parameter; count633++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count634 = 0; count634 < generation\_parameter\_x; count634++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count635 = 0; count635 < generation\_parameter\_y; count635++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count636 = 0; count636 < generation\_parameter\_z; count636++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count637 = 0; count637 < 10; count637++) {

for (var count638 = 0; count638 < 10; count638++) {

for (var count639 = 0; count639 < 10; count639++) {

hollow\_ingot();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 12;

density\_parameter\_x = 1.521920732e+25;

density\_parameter\_y = 4.150861668e+25;

density\_parameter\_z = 2.636057052e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count640 = 0; count640 < generation\_parameter\_x; count640++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count641 = 0; count641 < generation\_parameter\_y; count641++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count642 = 0; count642 < generation\_parameter\_z; count642++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 9.964110021e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 12;

for (var count643 = 0; count643 < generation\_parameter\_x; count643++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count644 = 0; count644 < generation\_parameter\_y; count644++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count645 = 0; count645 < generation\_parameter\_z; count645++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 1.177253179e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 13;

for (var count646 = 0; count646 < generation\_parameter\_x; count646++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count647 = 0; count647 < generation\_parameter\_y; count647++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count648 = 0; count648 < generation\_parameter\_z; count648++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1z = cursor1z + 1.757371368e+25;

sorter = 13;

for (var count649 = 0; count649 < generation\_parameter\_x; count649++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count650 = 0; count650 < generation\_parameter\_y; count650++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count651 = 0; count651 < generation\_parameter\_z; count651++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 12;

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

for (var count652 = 0; count652 < generation\_parameter\_x; count652++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count653 = 0; count653 < generation\_parameter\_y; count653++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count654 = 0; count654 < generation\_parameter\_z; count654++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 9.964110021e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 12;

for (var count655 = 0; count655 < generation\_parameter\_x; count655++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count656 = 0; count656 < generation\_parameter\_y; count656++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count657 = 0; count657 < generation\_parameter\_z; count657++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 1.177253179e+24;

cursor1x = cursor1x - 4.393428421e+24;

sorter = 13;

for (var count658 = 0; count658 < generation\_parameter\_x; count658++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count659 = 0; count659 < generation\_parameter\_y; count659++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count660 = 0; count660 < generation\_parameter\_z; count660++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1y = cursor1y + density\_parameter\_y / 2;

cursor1z = cursor1z + density\_parameter\_z / 2;

}

cursor1z = cursor1z + 1.757371368e+25;

sorter = 13;

for (var count661 = 0; count661 < generation\_parameter\_x; count661++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count662 = 0; count662 < generation\_parameter\_y; count662++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count663 = 0; count663 < generation\_parameter\_z; count663++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

sort\_allotrope\_bar();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count664 = 0; count664 < 10; count664++) {

for (var count665 = 0; count665 < 10; count665++) {

for (var count665b = 0; count665b < 10; count665b++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count667 = 0; count667 < generation\_parameter\_x; count667++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count668 = 0; count668 < generation\_parameter\_y; count668++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count669 = 0; count669 < generation\_parameter\_z; count669++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

indigotine();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 6.187927353e+31);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 6.187927353e+31)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count700 = 0; count700 < generation\_parameter\_x; count700++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count700b = 0; count700b < generation\_parameter\_y; count700b++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count701 = 0; count701 < generation\_parameter\_z; count701++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count702 = 0; count702 < 10; count702++) {

for (var count703 = 0; count703 < 10; count703++) {

for (var count704 = 0; count704 < 10; count704++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count705 = 0; count705 < generation\_parameter\_x; count705++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count706 = 0; count706 < generation\_parameter\_y; count706++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count707 = 0; count707 < generation\_parameter\_z; count707++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

carmine();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 6.187927353e+31);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 6.187927353e+31)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count708 = 0; count708 < generation\_parameter\_x; count708++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count709 = 0; count709 < generation\_parameter\_y; count709++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count710 = 0; count710 < generation\_parameter\_z; count710++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count711 = 0; count711 < 10; count711++) {

for (var count712 = 0; count712 < 10; count712++) {

for (var count713 = 0; count713 < 10; count713++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187927353e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count714 = 0; count714 < generation\_parameter\_x; count714++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count715 = 0; count715 < generation\_parameter\_y; count715++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count716 = 0; count716 < generation\_parameter\_z; count716++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

quinoline\_yellow();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 6.187927353e+31);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 6.187927353e+31)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count717 = 0; count717 < generation\_parameter\_x; count717++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count718 = 0; count718 < generation\_parameter\_y; count718++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count719 = 0; count719 < generation\_parameter\_z; count719++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function house\_8() {

unused\_variable = 5;

set\_earth\_rotation();

create\_building();

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.1;

centerpoint\_y = centerpoint\_y - (3.093963676e+34 \* 2) \* 0.51;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.1;

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

crystal\_spheres();

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 3;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 4;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 5;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 7;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 10;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 12;

crystal\_spheres();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 15;

crystal\_spheres();

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 1) \* 8;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 1) \* 1;

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 100;

concave\_lenses();

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 50;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 20;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 10;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 5;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 2.5;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 2;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.75;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.5;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.25;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.25;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.1;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.05;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.025;

concave\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.01;

concave\_lenses();

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 1) \* 15;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 1) \* 1;

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 100;

convex\_lenses();

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 50;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 20;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 10;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 5;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 2.5;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 2;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.75;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.5;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.25;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.25;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.1;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.05;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.025;

convex\_lenses();

}

if (unused\_variable == 5) {

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 1) \* 1;

used\_radius\_layer\_earth = 6.187865474e+33 / 2;

l2rgen = 6.187865474e+33 / 1.01;

convex\_lenses();

}

}

}

function crystal\_spheres() {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

guider\_x = starter\_x - 6.187865474e+33 / 2;

guider\_y = starter\_y - 6.187865474e+33 / 2;

guider\_z = starter\_z + 6.187865474e+33 / 2;

}

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length > 2) {

l1 = list\_\_universe\_string\_entities\_\_\_default.length;

c1 = 0;

for (var count720 = 0; count720 < l1; count720++) {

if (c1 + 2 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(c1 + 1) - 1];

c1 = c1 + 1;

obtain\_string\_location();

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - string\_x), 2) + (Math.pow(Math.abs(guider\_y - string\_y), 2) + Math.pow(Math.abs(guider\_z - string\_z), 2)))) < used\_radius\_layer\_earth) {

list.splice(c1 - 1, 1);

c1 = c1 - 1;

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count721 = 0; count721 < generation\_parameter\_x; count721++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count722 = 0; count722 < generation\_parameter\_y; count722++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count723 = 0; count723 < generation\_parameter\_z; count723++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count724 = 0; count724 < generation\_parameter\_x; count724++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count725 = 0; count725 < generation\_parameter\_y; count725++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count726 = 0; count726 < generation\_parameter\_z; count726++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count727 = 0; count727 < generation\_parameter\_x; count727++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count728 = 0; count728 < generation\_parameter\_y; count728++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count729 = 0; count729 < generation\_parameter\_z; count729++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count730 = 0; count730 < generation\_parameter\_x; count730++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count731 = 0; count731 < generation\_parameter\_y; count731++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count732 = 0; count732 < generation\_parameter\_z; count732++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count733 = 0; count733 < generation\_parameter\_x; count733++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count734 = 0; count734 < generation\_parameter\_y; count734++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count735 = 0; count735 < generation\_parameter\_z; count735++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count736 = 0; count736 < generation\_parameter\_x; count736++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count737 = 0; count737 < generation\_parameter\_y; count737++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count738 = 0; count738 < generation\_parameter\_z; count738++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count739 = 0; count739 < generation\_parameter\_x; count739++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count740 = 0; count740 < generation\_parameter\_y; count740++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count741 = 0; count741 < generation\_parameter\_z; count741++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count742 = 0; count742 < generation\_parameter\_x; count742++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count743 = 0; count743 < generation\_parameter\_y; count743++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count744 = 0; count744 < generation\_parameter\_z; count744++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function concave\_lenses() {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

guider\_x = starter\_x - 6.187865474e+33 / 2;

guider\_y = starter\_y - 6.187865474e+33 / 2;

guider\_z = starter\_z + 6.187865474e+33 / 2;

generator\_x = guider\_x;

generator\_z = guider\_z;

generator\_y = l2rgen + guider\_y;

}

if (list\_\_universe\_string\_entities\_\_\_default.length > 2) {

l1 = list\_\_universe\_string\_entities\_\_\_default.length;

c1 = 0;

for (var count745 = 0; count745 < l1; count745++) {

if (c1 + 2 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(c1 + 1) - 1];

c1 = c1 + 1;

obtain\_string\_location();

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - string\_x), 2) + (Math.pow(Math.abs(generator\_y - string\_y), 2) + Math.pow(Math.abs(generator\_z - string\_z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - string\_x), 2) + (Math.pow(Math.abs(guider\_y - string\_y), 2) + Math.pow(Math.abs(guider\_z - string\_z), 2)))) < used\_radius\_layer\_earth) {

list.splice(c1 - 1, 1);

c1 = c1 - 1;

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count746 = 0; count746 < generation\_parameter\_x; count746++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count747 = 0; count747 < generation\_parameter\_y; count747++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count748 = 0; count748 < generation\_parameter\_z; count748++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count749 = 0; count749 < generation\_parameter\_x; count749++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count750 = 0; count750 < generation\_parameter\_y; count750++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count751 = 0; count751 < generation\_parameter\_z; count751++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count752 = 0; count752 < generation\_parameter\_x; count752++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count753 = 0; count753 < generation\_parameter\_y; count753++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count754 = 0; count754 < generation\_parameter\_z; count754++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count755 = 0; count755 < generation\_parameter\_x; count755++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count756 = 0; count756 < generation\_parameter\_y; count756++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count757 = 0; count757 < generation\_parameter\_z; count757++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count758 = 0; count758 < generation\_parameter\_x; count758++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count759 = 0; count759 < generation\_parameter\_y; count759++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count760 = 0; count760 < generation\_parameter\_z; count760++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count761 = 0; count761 < generation\_parameter\_x; count761++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count762 = 0; count762 < generation\_parameter\_y; count762++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count763 = 0; count763 < generation\_parameter\_z; count763++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count764 = 0; count764 < generation\_parameter\_x; count764++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count765 = 0; count765 < generation\_parameter\_y; count765++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count766 = 0; count766 < generation\_parameter\_z; count766++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count767 = 0; count767 < generation\_parameter\_x; count767++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count768 = 0; count768 < generation\_parameter\_y; count768++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count769 = 0; count769 < generation\_parameter\_z; count769++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) > used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function convex\_lenses() {

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

guider\_x = starter\_x - 6.187865474e+33 / 2;

guider\_y = starter\_y - 6.187865474e+33 / 2;

guider\_z = starter\_z + 6.187865474e+33 / 2;

generator\_x = guider\_x;

generator\_z = guider\_z;

generator\_y = l2rgen + guider\_y;

}

if (list\_\_universe\_string\_entities\_\_\_default.length > 2) {

l1 = list\_\_universe\_string\_entities\_\_\_default.length;

c1 = 0;

for (var count770 = 0; count770 < l1; count770++) {

if (c1 + 2 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(c1 + 1) - 1];

c1 = c1 + 1;

obtain\_string\_location();

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - string\_x), 2) + (Math.pow(Math.abs(generator\_y - string\_y), 2) + Math.pow(Math.abs(generator\_z - string\_z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - string\_x), 2) + (Math.pow(Math.abs(guider\_y - string\_y), 2) + Math.pow(Math.abs(guider\_z - string\_z), 2)))) < used\_radius\_layer\_earth) {

list.splice(c1 - 1, 1);

c1 = c1 - 1;

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count771 = 0; count771 < generation\_parameter\_x; count771++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count772 = 0; count772 < generation\_parameter\_y; count772++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count773 = 0; count773 < generation\_parameter\_z; count773++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count774 = 0; count774 < generation\_parameter\_x; count774++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count775 = 0; count775 < generation\_parameter\_y; count775++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count776 = 0; count776 < generation\_parameter\_z; count776++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count777 = 0; count777 < generation\_parameter\_x; count777++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count778 = 0; count778 < generation\_parameter\_y; count778++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count779 = 0; count779 < generation\_parameter\_z; count779++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count780 = 0; count780 < generation\_parameter\_x; count780++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count781 = 0; count781 < generation\_parameter\_y; count781++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count782 = 0; count782 < generation\_parameter\_z; count782++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count783 = 0; count783 < generation\_parameter\_x; count783++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count784 = 0; count784 < generation\_parameter\_y; count784++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count785 = 0; count785 < generation\_parameter\_z; count785++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count786 = 0; count786 < generation\_parameter\_x; count786++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count787 = 0; count787 < generation\_parameter\_y; count787++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count788 = 0; count788 < generation\_parameter\_z; count788++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count789 = 0; count789 < generation\_parameter\_x; count789++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count790 = 0; count790 < generation\_parameter\_y; count790++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count791 = 0; count791 < generation\_parameter\_z; count791++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(6.187865474e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.187865474e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+33 / density\_parameter\_z);

for (var count792 = 0; count792 < generation\_parameter\_x; count792++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count793 = 0; count793 < generation\_parameter\_y; count793++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count794 = 0; count794 < generation\_parameter\_z; count794++) {

cursor1z = cursor1z + density\_parameter\_z;

if (Math.sqrt(Math.abs(Math.pow(Math.abs(generator\_x - cursor1x), 2) + (Math.pow(Math.abs(generator\_y - cursor1y), 2) + Math.pow(Math.abs(generator\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs(guider\_x - cursor1x), 2) + (Math.pow(Math.abs(guider\_y - cursor1y), 2) + Math.pow(Math.abs(guider\_z - cursor1z), 2)))) < used\_radius\_layer\_earth) {

set\_earth\_rotation();

sort\_allotrope\_bar();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function house\_10() {

unused\_variable = 5;

create\_building();

set\_earth\_rotation();

centerpoint\_x = centerpoint\_x - 6.197865474e+34;

centerpoint\_y = centerpoint\_y - 6.197865474e+34;

centerpoint\_z = centerpoint\_z + 6.197865474e+34;

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count795 = 0; count795 < 2000; count795++) {

for (var count796 = 0; count796 < 2000; count796++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count797 = 0; count797 < string\_solar\_distance\_\_length\_parameter; count797++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 2) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(3e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count798 = 0; count798 < generation\_parameter\_x; count798++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count799 = 0; count799 < generation\_parameter\_y; count799++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count800 = 0; count800 < generation\_parameter\_z; count800++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 2;

}

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

starter\_y = starter\_y - 6.187865474e+31 / 2;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count801 = 0; count801 < 1000; count801++) {

for (var count802 = 0; count802 < 1000; count802++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count803 = 0; count803 < string\_solar\_distance\_\_length\_parameter; count803++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 1) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count804 = 0; count804 < generation\_parameter\_x; count804++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count805 = 0; count805 < generation\_parameter\_y; count805++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count806 = 0; count806 < generation\_parameter\_z; count806++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 1;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 1;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count807 = 0; count807 < 1000; count807++) {

for (var count808 = 0; count808 < 1000; count808++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count809 = 0; count809 < string\_solar\_distance\_\_length\_parameter; count809++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.5) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(1.2e+32 / density\_parameter\_x);

generation\_parameter\_y = Math.round(1.2e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count810 = 0; count810 < generation\_parameter\_x; count810++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count811 = 0; count811 < generation\_parameter\_y; count811++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count812 = 0; count812 < generation\_parameter\_z; count812++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.5;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.5;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count813 = 0; count813 < 1000; count813++) {

for (var count814 = 0; count814 < 1000; count814++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count815 = 0; count815 < string\_solar\_distance\_\_length\_parameter; count815++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.2) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(3e+32 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count816 = 0; count816 < generation\_parameter\_x; count816++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count817 = 0; count817 < generation\_parameter\_y; count817++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count818 = 0; count818 < generation\_parameter\_z; count818++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.2;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.2;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count819 = 0; count819 < 1000; count819++) {

for (var count820 = 0; count820 < 1000; count820++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count821 = 0; count821 < string\_solar\_distance\_\_length\_parameter; count821++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.1) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6e+32 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count822 = 0; count822 < generation\_parameter\_x; count822++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count823 = 0; count823 < generation\_parameter\_y; count823++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count824 = 0; count824 < generation\_parameter\_z; count824++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.1;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.1;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count825 = 0; count825 < 1000; count825++) {

for (var count826 = 0; count826 < 1000; count826++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count827 = 0; count827 < string\_solar\_distance\_\_length\_parameter; count827++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.05) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+33) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(1.2e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(1.2e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count828 = 0; count828 < generation\_parameter\_x; count828++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count829 = 0; count829 < generation\_parameter\_y; count829++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count830 = 0; count830 < generation\_parameter\_z; count830++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.05;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.05;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count831 = 0; count831 < 1000; count831++) {

for (var count832 = 0; count832 < 1000; count832++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count833 = 0; count833 < string\_solar\_distance\_\_length\_parameter; count833++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.02) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+33) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(3e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count834 = 0; count834 < generation\_parameter\_x; count834++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count835 = 0; count835 < generation\_parameter\_y; count835++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count836 = 0; count836 < generation\_parameter\_z; count836++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.02;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.02;

}

}

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count837 = 0; count837 < 1000; count837++) {

for (var count838 = 0; count838 < 1000; count838++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count839 = 0; count839 < string\_solar\_distance\_\_length\_parameter; count839++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6e+31 / 0.01) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+33) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6e+33 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count840 = 0; count840 < generation\_parameter\_x; count840++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count841 = 0; count841 < generation\_parameter\_y; count841++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count842 = 0; count842 < generation\_parameter\_z; count842++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 0.01;

}

starter\_x = starter\_x + (6.187865474e+31 \* 1000) \* 1;

starter\_y = starter\_y - 6.187865474e+31 / 0.01;

}

}

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function house\_7() {

unused\_variable = 5;

set\_earth\_rotation();

create\_building();

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+29);

}

for (var count843 = 0; count843 < 7; count843++) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count844 = 0; count844 < 10; count844++) {

for (var count845 = 0; count845 < 10; count845++) {

for (var count846 = 0; count846 < 10; count846++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 0);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count847 = 0; count847 < generation\_parameter\_x; count847++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count848 = 0; count848 < generation\_parameter\_y; count848++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count849 = 0; count849 < generation\_parameter\_z; count849++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* 10.5;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

for (var count850 = 0; count850 < 7; count850++) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count851 = 0; count851 < 10; count851++) {

for (var count852 = 0; count852 < 10; count852++) {

for (var count853 = 0; count853 < 10; count853++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 0);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = 1.922068735e+25;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count854 = 0; count854 < generation\_parameter\_x; count854++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count855 = 0; count855 < generation\_parameter\_y; count855++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count856 = 0; count856 < generation\_parameter\_z; count856++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

water\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* 10.5;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

for (var count857 = 0; count857 < 7; count857++) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count858 = 0; count858 < 10; count858++) {

for (var count859 = 0; count859 < 10; count859++) {

for (var count860 = 0; count860 < 10; count860++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 0);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count861 = 0; count861 < generation\_parameter\_x; count861++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count862 = 0; count862 < generation\_parameter\_y; count862++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count863 = 0; count863 < generation\_parameter\_z; count863++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - (6.187865474e+32 + (6.187865474e+33 - 6.187865474e+32 \* 2) / 3);

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count864 = 0; count864 < generation\_parameter\_x; count864++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count865 = 0; count865 < generation\_parameter\_y; count865++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count866 = 0; count866 < generation\_parameter\_z; count866++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - (6.187865474e+32 + (6.187865474e+33 - 6.187865474e+32 \* 2) / 3) \* 2;

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count867 = 0; count867 < generation\_parameter\_x; count867++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count868 = 0; count868 < generation\_parameter\_y; count868++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count869 = 0; count869 < generation\_parameter\_z; count869++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* 10.5;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

for (var count870 = 0; count870 < 7; count870++) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count871 = 0; count871 < 10; count871++) {

for (var count872 = 0; count872 < 10; count872++) {

for (var count873 = 0; count873 < 10; count873++) {

container();

carve\_container();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 0);

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count874 = 0; count874 < generation\_parameter\_x; count874++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count875 = 0; count875 < generation\_parameter\_y; count875++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count876 = 0; count876 < generation\_parameter\_z; count876++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_1();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - (6.187865474e+32 + (6.187865474e+33 - 6.187865474e+32 \* 2) / 3);

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count877 = 0; count877 < generation\_parameter\_x; count877++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count878 = 0; count878 < generation\_parameter\_y; count878++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count879 = 0; count879 < generation\_parameter\_z; count879++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

guider\_x = starter\_x - (6.187865474e+32 + (6.187865474e+33 - 6.187865474e+32 \* 2) / 3) \* 2;

guider\_y = starter\_y - 6.187865474e+32;

guider\_z = starter\_z + 6.187865474e+32;

}

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 6.187927353e+25;

density\_parameter\_y = 6.187927353e+25;

density\_parameter\_z = 6.187927353e+25;

generation\_parameter\_x = Math.round(((6.187865474e+33 - 6.187865474e+32 \* 2) / 3) / density\_parameter\_x);

generation\_parameter\_y = Math.round((6.187865474e+33 - (6.187865474e+32 \* 2 + 0)) / density\_parameter\_y);

generation\_parameter\_z = Math.round((6.187865474e+33 - 6.187865474e+32 \* 2) / density\_parameter\_z);

for (var count880 = 0; count880 < generation\_parameter\_x; count880++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count881 = 0; count881 < generation\_parameter\_y; count881++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count882 = 0; count882 < generation\_parameter\_z; count882++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* 10.5;

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* 1.5;

for (var count883 = 0; count883 < 7; count883++) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count884 = 0; count884 < 10; count884++) {

for (var count885 = 0; count885 < 10; count885++) {

for (var count886 = 0; count886 < 10; count886++) {

container();

carve\_container();

if (unused\_variable == 5) {

guider\_x = starter\_x - 6.187865474e+32;

guider\_y = starter\_y - (6.187865474e+32 + 0);

guider\_z = starter\_z + 6.187865474e+32;

}

var repeat\_end887 = Math.floor((6.187865474e+33 - 6.187865474e+32 \* 2) / 6.187927353e+31) - 1;

for (var count887 = 0; count887 < repeat\_end887; count887++) {

guider\_x = guider\_x - 6.187927353e+31;

var repeat\_end888 = Math.floor((6.187865474e+33 - 6.187865474e+32 \* 2) / 6.187927353e+31) - 1;

for (var count888 = 0; count888 < repeat\_end888; count888++) {

guider\_y = guider\_y - 6.187927353e+31;

var repeat\_end889 = Math.floor((6.187865474e+33 - 6.187865474e+32 \* 2) / 6.187927353e+31) - 1;

for (var count889 = 0; count889 < repeat\_end889; count889++) {

guider\_z = guider\_z + 6.187927353e+31;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

for (var count890 = 0; count890 < generation\_parameter\_x; count890++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count891 = 0; count891 < generation\_parameter\_y; count891++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count892 = 0; count892 < generation\_parameter\_z; count892++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_chloride();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

for (var count893 = 0; count893 < generation\_parameter\_x; count893++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count894 = 0; count894 < generation\_parameter\_y; count894++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count895 = 0; count895 < generation\_parameter\_z; count895++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_chloride();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1z = cursor1z + density\_parameter\_x / 2;

for (var count896 = 0; count896 < generation\_parameter\_x; count896++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count897 = 0; count897 < generation\_parameter\_y; count897++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count898 = 0; count898 < generation\_parameter\_z; count898++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_chloride();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1y = cursor1y + density\_parameter\_x / 2;

cursor1z = cursor1z + density\_parameter\_x / 2;

for (var count899 = 0; count899 < generation\_parameter\_x; count899++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count900 = 0; count900 < generation\_parameter\_y; count900++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count901 = 0; count901 < generation\_parameter\_z; count901++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_chloride();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1y = cursor1y + density\_parameter\_x / 2;

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1x = cursor1x + density\_parameter\_x / 2;

for (var count902 = 0; count902 < generation\_parameter\_x; count902++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count903 = 0; count903 < generation\_parameter\_y; count903++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count904 = 0; count904 < generation\_parameter\_z; count904++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_sodium();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1y = cursor1y + density\_parameter\_x / 2;

for (var count905 = 0; count905 < generation\_parameter\_x; count905++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count906 = 0; count906 < generation\_parameter\_y; count906++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count907 = 0; count907 < generation\_parameter\_z; count907++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_sodium();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1z = cursor1z + density\_parameter\_x / 2;

for (var count908 = 0; count908 < generation\_parameter\_x; count908++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count909 = 0; count909 < generation\_parameter\_y; count909++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count910 = 0; count910 < generation\_parameter\_z; count910++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_sodium();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = guider\_x;

cursor1y = guider\_y;

cursor1z = guider\_z;

}

sorter = 12;

density\_parameter\_x = 3.489991027e+25;

density\_parameter\_y = 3.489991027e+25;

density\_parameter\_z = 3.489991027e+25;

generation\_parameter\_x = Math.round(6.181739426e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6.181739426e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.181739426e+31 / density\_parameter\_z);

cursor1x = cursor1x + density\_parameter\_x / 2;

for (var count911 = 0; count911 < generation\_parameter\_x; count911++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count912 = 0; count912 < generation\_parameter\_y; count912++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count913 = 0; count913 < generation\_parameter\_z; count913++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

salt\_sodium();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

guider\_z = guider\_z - (Math.floor((6.187865474e+33 - 6.187865474e+32 \* 2) / 6.187927353e+31) - 1) \* 6.187927353e+31;

}

guider\_y = guider\_y + (Math.floor((6.187865474e+33 - 6.187865474e+32 \* 2) / 6.187927353e+31) - 1) \* 6.187927353e+31;

}

starter\_z = starter\_z + 6.187927353e+33;

}

starter\_z = starter\_z - 6.187927353e+34;

starter\_x = starter\_x - 6.187927353e+33;

}

starter\_x = starter\_x + 6.187927353e+34;

starter\_y = starter\_y - 6.187927353e+33;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

}

function house\_11() {

unused\_variable = 5;

create\_building();

set\_earth\_rotation();

centerpoint\_x = centerpoint\_x - 6.197865474e+34;

centerpoint\_y = centerpoint\_y - 6.197865474e+34;

centerpoint\_z = centerpoint\_z + 6.197865474e+34;

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count914 = 0; count914 < 2000; count914++) {

for (var count915 = 0; count915 < 2000; count915++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count916 = 0; count916 < string\_solar\_distance\_\_length\_parameter; count916++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 2e+31 / 1) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 2e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(2e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(2e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count917 = 0; count917 < generation\_parameter\_x; count917++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count918 = 0; count918 < generation\_parameter\_y; count918++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count919 = 0; count919 < generation\_parameter\_z; count919++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 2;

}

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

starter\_y = starter\_y - 6.187865474e+31 / 2;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

for (var count920 = 0; count920 < 1000; count920++) {

if (unused\_variable == 5) {

for (var count921 = 0; count921 < 1000; count921++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count922 = 0; count922 < string\_solar\_distance\_\_length\_parameter; count922++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 2e+31 / 1) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 2e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(2e+31 / density\_parameter\_x);

generation\_parameter\_y = Math.round(2e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count923 = 0; count923 < generation\_parameter\_x; count923++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count924 = 0; count924 < generation\_parameter\_y; count924++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count925 = 0; count925 < generation\_parameter\_z; count925++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_x = starter\_x - 6.187865474e+31 / 1;

}

starter\_x = starter\_x + 6.187865474e+31 \* 1000;

starter\_y = starter\_y - 6.187865474e+30 / 1;

for (var count926 = 0; count926 < 1000; count926++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count927 = 0; count927 < string\_solar\_distance\_\_length\_parameter; count927++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 1e+31 / 1) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(1e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(1e+31 / density\_parameter\_z);

for (var count928 = 0; count928 < generation\_parameter\_x; count928++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count929 = 0; count929 < generation\_parameter\_y; count929++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count930 = 0; count930 < generation\_parameter\_z; count930++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_z = starter\_z + 6.187865474e+31 / 1;

}

starter\_z = starter\_z - 6.187865474e+31 \* 1000;

starter\_y = starter\_y + 6.187865474e+30 / 1;

starter\_y = starter\_y - 6.187865474e+31 / 1;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

needles();

}

function needles() {

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count931 = 0; count931 < 10; count931++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count932 = 0; count932 < 10; count932++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((1e+32 / 1) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end933 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count933 = 0; count933 < repeat\_end933; count933++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_z > starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 10) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

} else if (string\_x > starter\_x - (used\_density\_parameter \* generation\_parameter\_z) / 40) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_x < (starter\_x - used\_density\_parameter \* generation\_parameter\_z) + (used\_density\_parameter \* generation\_parameter\_z) / 40) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count934 = 0; count934 < generation\_parameter\_x; count934++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count935 = 0; count935 < generation\_parameter\_y; count935++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count936 = 0; count936 < generation\_parameter\_z; count936++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 10) {

if (Math.abs(starter\_x - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

} else if (cursor1x > starter\_x - (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x < (starter\_x - used\_density\_parameter \* generation\_parameter\_z) + (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count937 = 0; count937 < generation\_parameter\_x; count937++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count938 = 0; count938 < generation\_parameter\_y; count938++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count939 = 0; count939 < generation\_parameter\_z; count939++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 10) {

if (Math.abs(starter\_x - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

} else if (cursor1x > starter\_x - (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x < (starter\_x - used\_density\_parameter \* generation\_parameter\_z) + (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count940 = 0; count940 < generation\_parameter\_x; count940++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count941 = 0; count941 < generation\_parameter\_y; count941++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count942 = 0; count942 < generation\_parameter\_z; count942++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 10) {

if (Math.abs(starter\_x - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

} else if (cursor1x > starter\_x - (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x < (starter\_x - used\_density\_parameter \* generation\_parameter\_z) + (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count943 = 0; count943 < generation\_parameter\_x; count943++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count944 = 0; count944 < generation\_parameter\_y; count944++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count945 = 0; count945 < generation\_parameter\_z; count945++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 10) {

if (Math.abs(starter\_x - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 1) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

} else if (cursor1x > starter\_x - (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x < (starter\_x - used\_density\_parameter \* generation\_parameter\_z) + (used\_density\_parameter \* generation\_parameter\_z) / 40) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function house\_9() {

unused\_variable = 5;

create\_building();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 + 3.093963676e+32);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+32);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+32);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count946 = 0; count946 < 100; count946++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end947 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count947 = 0; count947 < repeat\_end947; count947++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 3e+34) {

if (string\_z > starter\_z) {

if (string\_x > starter\_x - 6.187865474e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - string\_y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(3e+34 / used\_density\_parameter);

for (var count948 = 0; count948 < generation\_parameter\_x; count948++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count949 = 0; count949 < generation\_parameter\_y; count949++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count950 = 0; count950 < generation\_parameter\_z; count950++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count951 = 0; count951 < generation\_parameter\_x; count951++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count952 = 0; count952 < generation\_parameter\_y; count952++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count953 = 0; count953 < generation\_parameter\_z; count953++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count954 = 0; count954 < generation\_parameter\_x; count954++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count955 = 0; count955 < generation\_parameter\_y; count955++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count956 = 0; count956 < generation\_parameter\_z; count956++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count957 = 0; count957 < generation\_parameter\_x; count957++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count958 = 0; count958 < generation\_parameter\_y; count958++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count959 = 0; count959 < generation\_parameter\_z; count959++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count960 = 0; count960 < 100; count960++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end961 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count961 = 0; count961 < repeat\_end961; count961++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 3e+34) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + 6.187865474e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - string\_y) > Math.abs(string\_z - (starter\_z + 3e+34)) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

} else if (string\_x > starter\_x - 3.3e+33) {

if (string\_x < starter\_x - 2.7e+33) {

if (string\_z < starter\_z + 2.42e+34) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(3e+34 / used\_density\_parameter);

for (var count962 = 0; count962 < generation\_parameter\_x; count962++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count963 = 0; count963 < generation\_parameter\_y; count963++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count964 = 0; count964 < generation\_parameter\_z; count964++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1z - (starter\_z + 3e+34)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1x > starter\_x - 3.3e+33) {

if (cursor1z < starter\_x - 2.7e+33) {

if (cursor1z < starter\_z + 2.42e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count965 = 0; count965 < generation\_parameter\_x; count965++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count966 = 0; count966 < generation\_parameter\_y; count966++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count967 = 0; count967 < generation\_parameter\_z; count967++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1z - (starter\_z + 3e+34)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1x > starter\_x - 3.3e+33) {

if (cursor1z < starter\_x - 2.7e+33) {

if (cursor1z < starter\_z + 2.42e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count968 = 0; count968 < generation\_parameter\_x; count968++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count969 = 0; count969 < generation\_parameter\_y; count969++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count970 = 0; count970 < generation\_parameter\_z; count970++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1z - (starter\_z + 3e+34)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1x > starter\_x - 3.3e+33) {

if (cursor1z < starter\_x - 2.7e+33) {

if (cursor1z < starter\_z + 2.42e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count971 = 0; count971 < generation\_parameter\_x; count971++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count972 = 0; count972 < generation\_parameter\_y; count972++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count973 = 0; count973 < generation\_parameter\_z; count973++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 2.4e+34) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1z - (starter\_z + 3e+34)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1x > starter\_x - 3.3e+33) {

if (cursor1z < starter\_x - 2.7e+33) {

if (cursor1z < starter\_z + 2.42e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

house\_9b();

}

}

function house\_9b() {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count974 = 0; count974 < 100; count974++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end975 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count975 = 0; count975 < repeat\_end975; count975++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 3e+34) {

if (string\_z > starter\_z) {

if (string\_x > starter\_x - 6.187865474e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_z > starter\_z + 1.3e+34) {

if (Math.abs(starter\_y - string\_y) > 3e+31) {

if (Math.abs(starter\_y - string\_y) > Math.abs(string\_x - (starter\_x - 6.187865474e+33)) / 4) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(3e+34 / used\_density\_parameter);

for (var count976 = 0; count976 < generation\_parameter\_x; count976++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count977 = 0; count977 < generation\_parameter\_y; count977++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count978 = 0; count978 < generation\_parameter\_z; count978++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 1.3e+34) {

if (Math.abs(starter\_y - cursor1y) > 3e+31) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1x - (starter\_x - 6.187865474e+33)) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count979 = 0; count979 < generation\_parameter\_x; count979++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count980 = 0; count980 < generation\_parameter\_y; count980++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count981 = 0; count981 < generation\_parameter\_z; count981++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 1.3e+34) {

if (Math.abs(starter\_y - cursor1y) > 3e+31) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1x - (starter\_x - 6.187865474e+33)) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count982 = 0; count982 < generation\_parameter\_x; count982++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count983 = 0; count983 < generation\_parameter\_y; count983++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count984 = 0; count984 < generation\_parameter\_z; count984++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 1.3e+34) {

if (Math.abs(starter\_y - cursor1y) > 3e+31) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1x - (starter\_x - 6.187865474e+33)) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count985 = 0; count985 < generation\_parameter\_x; count985++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count986 = 0; count986 < generation\_parameter\_y; count986++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count987 = 0; count987 < generation\_parameter\_z; count987++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1x > starter\_x - 6.187865474e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1z > starter\_z + 1.3e+34) {

if (Math.abs(starter\_y - cursor1y) > 3e+31) {

if (Math.abs(starter\_y - cursor1y) > Math.abs(cursor1x - (starter\_x - 6.187865474e+33)) / 4) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count988 = 0; count988 < 9; count988++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end989 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count989 = 0; count989 < repeat\_end989; count989++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 100) {

if (string\_z < starter\_z + 3e+34) {

if (string\_z > starter\_z) {

if (string\_z > starter\_z + 2.4e+34) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_x > starter\_x - 3.7e+33) {

if (string\_x < starter\_x - 2.3e+33) {

if (starter\_y < starter\_y - 2.3e+33) {

if (starter\_y > starter\_y - 3.7e+33) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 100) / used\_density\_parameter);

generation\_parameter\_z = Math.round(3e+34 / used\_density\_parameter);

for (var count990 = 0; count990 < generation\_parameter\_x; count990++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count991 = 0; count991 < generation\_parameter\_y; count991++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count992 = 0; count992 < generation\_parameter\_z; count992++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z > starter\_z + 2.4e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x > starter\_x - 3.7e+33) {

if (cursor1x < starter\_x - 2.3e+33) {

if (cursor1y < starter\_y - 2.3e+33) {

if (cursor1y > starter\_y - 3.7e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count993 = 0; count993 < generation\_parameter\_x; count993++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count994 = 0; count994 < generation\_parameter\_y; count994++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count995 = 0; count995 < generation\_parameter\_z; count995++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z > starter\_z + 2.4e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x > starter\_x - 3.7e+33) {

if (cursor1x < starter\_x - 2.3e+33) {

if (cursor1y < starter\_y - 2.3e+33) {

if (cursor1y > starter\_y - 3.7e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count996 = 0; count996 < generation\_parameter\_x; count996++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count997 = 0; count997 < generation\_parameter\_y; count997++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count998 = 0; count998 < generation\_parameter\_z; count998++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z > starter\_z + 2.4e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x > starter\_x - 3.7e+33) {

if (cursor1x < starter\_x - 2.3e+33) {

if (cursor1y < starter\_y - 2.3e+33) {

if (cursor1y > starter\_y - 3.7e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count999 = 0; count999 < generation\_parameter\_x; count999++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1000 = 0; count1000 < generation\_parameter\_y; count1000++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1001 = 0; count1001 < generation\_parameter\_z; count1001++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z > starter\_z + 2.4e+34) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1x > starter\_x - 3.7e+33) {

if (cursor1x < starter\_x - 2.3e+33) {

if (cursor1y < starter\_y - 2.3e+33) {

if (cursor1y > starter\_y - 3.7e+33) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1002 = 0; count1002 < 100; count1002++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

guider\_x = starter\_x - 6.187865474e+32 / 1;

guider\_z = starter\_z + 1.5e+33;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1003 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1003 = 0; count1003 < repeat\_end1003; count1003++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 3e+33) {

if (string\_z > starter\_z) {

if (Math.sqrt(Math.abs(Math.pow(string\_z - guider\_z, 2) + Math.pow(string\_x - guider\_x, 2))) < 3e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (string\_y < starter\_y - 6.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(string\_z - guider\_z, 2) + Math.pow(string\_x - guider\_x, 2))) < 6e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

} else if (string\_y > starter\_y - 1e+32) {

if (Math.sqrt(Math.abs(Math.pow(string\_z - guider\_z, 2) + Math.pow(string\_x - guider\_x, 2))) < 6e+32) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

} else if (string\_y > starter\_y - 6e+32) {

if (string\_y < starter\_y - 4e+32) {

if (Math.sqrt(Math.abs(Math.pow(string\_z - guider\_z, 2) + Math.pow(string\_x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - string\_y) < Math.abs(starter\_x - string\_x) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

} else if (string\_y > starter\_y - 3.9999e+32) {

if (string\_y < starter\_y - 1.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(string\_z - guider\_z, 2) + Math.pow(string\_x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - string\_y) < Math.abs((starter\_x - 6.187865474e+33) - Math.abs(starter\_x - string\_x)) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(3e+33 / used\_density\_parameter);

for (var count1004 = 0; count1004 < generation\_parameter\_x; count1004++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1005 = 0; count1005 < generation\_parameter\_y; count1005++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1006 = 0; count1006 < generation\_parameter\_z; count1006++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 3e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 6.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 1e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 6e+32) {

if (cursor1y < starter\_y - 4e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs(starter\_x - cursor1y) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

} else if (cursor1y > starter\_y - 3.9999e+32) {

if (cursor1y < starter\_y - 1.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_x - 6.187865474e+33) - Math.abs(starter\_x - cursor1x)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1007 = 0; count1007 < generation\_parameter\_x; count1007++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1008 = 0; count1008 < generation\_parameter\_y; count1008++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1009 = 0; count1009 < generation\_parameter\_z; count1009++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 3e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 6.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 1e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 6e+32) {

if (cursor1y < starter\_y - 4e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs(starter\_x - cursor1y) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

} else if (cursor1y > starter\_y - 3.9999e+32) {

if (cursor1y < starter\_y - 1.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_x - 6.187865474e+33) - Math.abs(starter\_x - cursor1x)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1010 = 0; count1010 < generation\_parameter\_x; count1010++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1011 = 0; count1011 < generation\_parameter\_y; count1011++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1012 = 0; count1012 < generation\_parameter\_z; count1012++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 3e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 6.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 1e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 6e+32) {

if (cursor1y < starter\_y - 4e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs(starter\_x - cursor1y) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

} else if (cursor1y > starter\_y - 3.9999e+32) {

if (cursor1y < starter\_y - 1.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_x - 6.187865474e+33) - Math.abs(starter\_x - cursor1x)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1013 = 0; count1013 < generation\_parameter\_x; count1013++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1014 = 0; count1014 < generation\_parameter\_y; count1014++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1015 = 0; count1015 < generation\_parameter\_z; count1015++) {

cursor1z = cursor1z + used\_density\_parameter;

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 3e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (cursor1y < starter\_y - 6.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 1e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) < 6e+32) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

} else if (cursor1y > starter\_y - 6e+32) {

if (cursor1y < starter\_y - 4e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs(starter\_x - cursor1y) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

} else if (cursor1y > starter\_y - 3.9999e+32) {

if (cursor1y < starter\_y - 1.0001e+32) {

if (Math.sqrt(Math.abs(Math.pow(cursor1z - guider\_z, 2) + Math.pow(cursor1x - guider\_x, 2))) > 3.0001e+32) {

if (Math.abs(starter\_y - cursor1y) < Math.abs((starter\_x - 6.187865474e+33) - Math.abs(starter\_x - cursor1x)) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

}

function house\_5() {

unused\_variable = 5;

set\_earth\_rotation();

create\_building();

if (unused\_variable == 5) {

material\_temperature\_\_kelvins = 275.18;

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+29);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1016 = 0; count1016 < 1000; count1016++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1017 = 0; count1017 < string\_solar\_distance\_\_length\_parameter; count1017++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1018 = 0; count1018 < generation\_parameter\_x; count1018++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1019 = 0; count1019 < generation\_parameter\_y; count1019++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1020 = 0; count1020 < generation\_parameter\_z; count1020++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1021 = 0; count1021 < 1000; count1021++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1022 = 0; count1022 < string\_solar\_distance\_\_length\_parameter; count1022++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+31) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+31 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1023 = 0; count1023 < generation\_parameter\_x; count1023++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1024 = 0; count1024 < generation\_parameter\_y; count1024++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1025 = 0; count1025 < generation\_parameter\_z; count1025++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1026 = 0; count1026 < 500; count1026++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1027 = 0; count1027 < string\_solar\_distance\_\_length\_parameter; count1027++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(1.2e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1028 = 0; count1028 < generation\_parameter\_x; count1028++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1029 = 0; count1029 < generation\_parameter\_y; count1029++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1030 = 0; count1030 < generation\_parameter\_z; count1030++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31 \* 2;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1031 = 0; count1031 < 200; count1031++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1032 = 0; count1032 < string\_solar\_distance\_\_length\_parameter; count1032++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1033 = 0; count1033 < generation\_parameter\_x; count1033++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1034 = 0; count1034 < generation\_parameter\_y; count1034++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1035 = 0; count1035 < generation\_parameter\_z; count1035++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31 \* 5;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1036 = 0; count1036 < 100; count1036++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1037 = 0; count1037 < string\_solar\_distance\_\_length\_parameter; count1037++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6e+32) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(6e+32 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1038 = 0; count1038 < generation\_parameter\_x; count1038++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1039 = 0; count1039 < generation\_parameter\_y; count1039++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1040 = 0; count1040 < generation\_parameter\_z; count1040++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31 \* 10;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1041 = 0; count1041 < 50; count1041++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1042 = 0; count1042 < string\_solar\_distance\_\_length\_parameter; count1042++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+33) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(1.2e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1043 = 0; count1043 < generation\_parameter\_x; count1043++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1044 = 0; count1044 < generation\_parameter\_y; count1044++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1045 = 0; count1045 < generation\_parameter\_z; count1045++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31 \* 20;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1046 = 0; count1046 < 20; count1046++) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

for (var count1047 = 0; count1047 < string\_solar\_distance\_\_length\_parameter; count1047++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 3e+33) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z - 6.187865474e+34) {

list\_\_universe\_string\_entities\_\_\_default.splice((string\_solar\_\_distance\_parameter + 0) - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

sorter = 11;

density\_parameter\_x = 5.086476284e+25;

density\_parameter\_y = 4.826583335e+25;

density\_parameter\_z = 6.435444447e+25;

generation\_parameter\_x = Math.round(6.187865474e+34 / density\_parameter\_x);

generation\_parameter\_y = Math.round(3e+33 / density\_parameter\_y);

generation\_parameter\_z = Math.round(6.187865474e+34 / density\_parameter\_z);

for (var count1048 = 0; count1048 < generation\_parameter\_x; count1048++) {

cursor1x = cursor1x - density\_parameter\_x;

for (var count1049 = 0; count1049 < generation\_parameter\_y; count1049++) {

cursor1y = cursor1y - density\_parameter\_y;

for (var count1050 = 0; count1050 < generation\_parameter\_z; count1050++) {

cursor1z = cursor1z + density\_parameter\_z;

cellulose();

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

starter\_y = starter\_y - 6.187865474e+31 \* 50;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function house\_4() {

create\_building();

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

centerpoint\_y = centerpoint\_y - (3.093963676e+34 + 3.093963676e+32);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2 + 3.093963676e+32);

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2 + 3.093963676e+32);

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1051 = 0; count1051 < 39990; count1051++) {

starter\_y = starter\_y - 1.546981838e+30;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1052 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1052 = 0; count1052 < repeat\_end1052; count1052++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.445e+30) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1053 = 0; count1053 < generation\_parameter\_x; count1053++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1054 = 0; count1054 < generation\_parameter\_y; count1054++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1055 = 0; count1055 < generation\_parameter\_z; count1055++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1056 = 0; count1056 < generation\_parameter\_x; count1056++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1057 = 0; count1057 < generation\_parameter\_y; count1057++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1058 = 0; count1058 < generation\_parameter\_z; count1058++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1059 = 0; count1059 < generation\_parameter\_x; count1059++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1060 = 0; count1060 < generation\_parameter\_y; count1060++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1061 = 0; count1061 < generation\_parameter\_z; count1061++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1062 = 0; count1062 < generation\_parameter\_x; count1062++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1063 = 0; count1063 < generation\_parameter\_y; count1063++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1064 = 0; count1064 < generation\_parameter\_z; count1064++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1065 = 0; count1065 < 5000; count1065++) {

starter\_y = starter\_y - 1.23758547e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1066 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1066 = 0; count1066 < repeat\_end1066; count1066++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1067 = 0; count1067 < generation\_parameter\_x; count1067++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1068 = 0; count1068 < generation\_parameter\_y; count1068++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1069 = 0; count1069 < generation\_parameter\_z; count1069++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1070 = 0; count1070 < generation\_parameter\_x; count1070++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1071 = 0; count1071 < generation\_parameter\_y; count1071++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1072 = 0; count1072 < generation\_parameter\_z; count1072++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1073 = 0; count1073 < generation\_parameter\_x; count1073++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1074 = 0; count1074 < generation\_parameter\_y; count1074++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1075 = 0; count1075 < generation\_parameter\_z; count1075++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1076 = 0; count1076 < generation\_parameter\_x; count1076++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1077 = 0; count1077 < generation\_parameter\_y; count1077++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1078 = 0; count1078 < generation\_parameter\_z; count1078++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1079 = 0; count1079 < 1000; count1079++) {

starter\_y = starter\_y - 6.187927353e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1080 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1080 = 0; count1080 < repeat\_end1080; count1080++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1081 = 0; count1081 < generation\_parameter\_x; count1081++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1082 = 0; count1082 < generation\_parameter\_y; count1082++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1083 = 0; count1083 < generation\_parameter\_z; count1083++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1084 = 0; count1084 < generation\_parameter\_x; count1084++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1085 = 0; count1085 < generation\_parameter\_y; count1085++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1086 = 0; count1086 < generation\_parameter\_z; count1086++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1087 = 0; count1087 < generation\_parameter\_x; count1087++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1088 = 0; count1088 < generation\_parameter\_y; count1088++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1089 = 0; count1089 < generation\_parameter\_z; count1089++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1090 = 0; count1090 < generation\_parameter\_x; count1090++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1091 = 0; count1091 < generation\_parameter\_y; count1091++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1092 = 0; count1092 < generation\_parameter\_z; count1092++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1093 = 0; count1093 < 500; count1093++) {

starter\_y = starter\_y - 6.187927353e+31 \* 2;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1094 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1094 = 0; count1094 < repeat\_end1094; count1094++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 2) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1095 = 0; count1095 < generation\_parameter\_x; count1095++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1096 = 0; count1096 < generation\_parameter\_y; count1096++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1097 = 0; count1097 < generation\_parameter\_z; count1097++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1098 = 0; count1098 < generation\_parameter\_x; count1098++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1099 = 0; count1099 < generation\_parameter\_y; count1099++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1100 = 0; count1100 < generation\_parameter\_z; count1100++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1101 = 0; count1101 < generation\_parameter\_x; count1101++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1102 = 0; count1102 < generation\_parameter\_y; count1102++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1103 = 0; count1103 < generation\_parameter\_z; count1103++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1104 = 0; count1104 < generation\_parameter\_x; count1104++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1105 = 0; count1105 < generation\_parameter\_y; count1105++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1106 = 0; count1106 < generation\_parameter\_z; count1106++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1107 = 0; count1107 < 200; count1107++) {

starter\_y = starter\_y - 6.187927353e+31 \* 5;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1108 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1108 = 0; count1108 < repeat\_end1108; count1108++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 5) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1109 = 0; count1109 < generation\_parameter\_x; count1109++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1110 = 0; count1110 < generation\_parameter\_y; count1110++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1111 = 0; count1111 < generation\_parameter\_z; count1111++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1112 = 0; count1112 < generation\_parameter\_x; count1112++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1113 = 0; count1113 < generation\_parameter\_y; count1113++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1114 = 0; count1114 < generation\_parameter\_z; count1114++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1115 = 0; count1115 < generation\_parameter\_x; count1115++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1116 = 0; count1116 < generation\_parameter\_y; count1116++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1117 = 0; count1117 < generation\_parameter\_z; count1117++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1118 = 0; count1118 < generation\_parameter\_x; count1118++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1119 = 0; count1119 < generation\_parameter\_y; count1119++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1120 = 0; count1120 < generation\_parameter\_z; count1120++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1121 = 0; count1121 < 100; count1121++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1122 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1122 = 0; count1122 < repeat\_end1122; count1122++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1123 = 0; count1123 < generation\_parameter\_x; count1123++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1124 = 0; count1124 < generation\_parameter\_y; count1124++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1125 = 0; count1125 < generation\_parameter\_z; count1125++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1126 = 0; count1126 < generation\_parameter\_x; count1126++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1127 = 0; count1127 < generation\_parameter\_y; count1127++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1128 = 0; count1128 < generation\_parameter\_z; count1128++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1129 = 0; count1129 < generation\_parameter\_x; count1129++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1130 = 0; count1130 < generation\_parameter\_y; count1130++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1131 = 0; count1131 < generation\_parameter\_z; count1131++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1132 = 0; count1132 < generation\_parameter\_x; count1132++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1133 = 0; count1133 < generation\_parameter\_y; count1133++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1134 = 0; count1134 < generation\_parameter\_z; count1134++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* (5 \* 1.5);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1135 = 0; count1135 < 39990; count1135++) {

starter\_y = starter\_y - 1.546981838e+30;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1136 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1136 = 0; count1136 < repeat\_end1136; count1136++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.445e+30) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1137 = 0; count1137 < generation\_parameter\_x; count1137++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1138 = 0; count1138 < generation\_parameter\_y; count1138++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1139 = 0; count1139 < generation\_parameter\_z; count1139++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1140 = 0; count1140 < generation\_parameter\_x; count1140++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1141 = 0; count1141 < generation\_parameter\_y; count1141++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1142 = 0; count1142 < generation\_parameter\_z; count1142++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1143 = 0; count1143 < generation\_parameter\_x; count1143++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1144 = 0; count1144 < generation\_parameter\_y; count1144++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1145 = 0; count1145 < generation\_parameter\_z; count1145++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1146 = 0; count1146 < generation\_parameter\_x; count1146++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1147 = 0; count1147 < generation\_parameter\_y; count1147++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1148 = 0; count1148 < generation\_parameter\_z; count1148++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1149 = 0; count1149 < 5000; count1149++) {

starter\_y = starter\_y - 1.23758547e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1150 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1150 = 0; count1150 < repeat\_end1150; count1150++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1151 = 0; count1151 < generation\_parameter\_x; count1151++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1152 = 0; count1152 < generation\_parameter\_y; count1152++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1153 = 0; count1153 < generation\_parameter\_z; count1153++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1154 = 0; count1154 < generation\_parameter\_x; count1154++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1155 = 0; count1155 < generation\_parameter\_y; count1155++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1156 = 0; count1156 < generation\_parameter\_z; count1156++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1157 = 0; count1157 < generation\_parameter\_x; count1157++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1158 = 0; count1158 < generation\_parameter\_y; count1158++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1159 = 0; count1159 < generation\_parameter\_z; count1159++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1160 = 0; count1160 < generation\_parameter\_x; count1160++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1161 = 0; count1161 < generation\_parameter\_y; count1161++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1162 = 0; count1162 < generation\_parameter\_z; count1162++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1163 = 0; count1163 < 1000; count1163++) {

starter\_y = starter\_y - 6.187927353e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1164 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1164 = 0; count1164 < repeat\_end1164; count1164++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1165 = 0; count1165 < generation\_parameter\_x; count1165++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1166 = 0; count1166 < generation\_parameter\_y; count1166++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1167 = 0; count1167 < generation\_parameter\_z; count1167++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1168 = 0; count1168 < generation\_parameter\_x; count1168++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1169 = 0; count1169 < generation\_parameter\_y; count1169++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1170 = 0; count1170 < generation\_parameter\_z; count1170++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1171 = 0; count1171 < generation\_parameter\_x; count1171++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1172 = 0; count1172 < generation\_parameter\_y; count1172++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1173 = 0; count1173 < generation\_parameter\_z; count1173++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1174 = 0; count1174 < generation\_parameter\_x; count1174++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1175 = 0; count1175 < generation\_parameter\_y; count1175++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1176 = 0; count1176 < generation\_parameter\_z; count1176++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1177 = 0; count1177 < 500; count1177++) {

starter\_y = starter\_y - 6.187927353e+31 \* 2;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1178 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1178 = 0; count1178 < repeat\_end1178; count1178++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 2) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1179 = 0; count1179 < generation\_parameter\_x; count1179++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1180 = 0; count1180 < generation\_parameter\_y; count1180++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1181 = 0; count1181 < generation\_parameter\_z; count1181++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1182 = 0; count1182 < generation\_parameter\_x; count1182++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1183 = 0; count1183 < generation\_parameter\_y; count1183++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1184 = 0; count1184 < generation\_parameter\_z; count1184++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1185 = 0; count1185 < generation\_parameter\_x; count1185++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1186 = 0; count1186 < generation\_parameter\_y; count1186++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1187 = 0; count1187 < generation\_parameter\_z; count1187++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1188 = 0; count1188 < generation\_parameter\_x; count1188++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1189 = 0; count1189 < generation\_parameter\_y; count1189++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1190 = 0; count1190 < generation\_parameter\_z; count1190++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_y = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1191 = 0; count1191 < 200; count1191++) {

starter\_y = starter\_y - 6.187927353e+31 \* 5;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1192 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1192 = 0; count1192 < repeat\_end1192; count1192++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 5) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1193 = 0; count1193 < generation\_parameter\_x; count1193++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1194 = 0; count1194 < generation\_parameter\_y; count1194++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1195 = 0; count1195 < generation\_parameter\_z; count1195++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1196 = 0; count1196 < generation\_parameter\_x; count1196++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1197 = 0; count1197 < generation\_parameter\_y; count1197++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1198 = 0; count1198 < generation\_parameter\_z; count1198++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1199 = 0; count1199 < generation\_parameter\_x; count1199++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1200 = 0; count1200 < generation\_parameter\_y; count1200++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1201 = 0; count1201 < generation\_parameter\_z; count1201++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1202 = 0; count1202 < generation\_parameter\_x; count1202++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1203 = 0; count1203 < generation\_parameter\_y; count1203++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1204 = 0; count1204 < generation\_parameter\_z; count1204++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1205 = 0; count1205 < 100; count1205++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1205b = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1205b = 0; count1205b < repeat\_end1205b; count1205b++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+34) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.187927353e+34 / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1206 = 0; count1206 < generation\_parameter\_x; count1206++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1207 = 0; count1207 < generation\_parameter\_y; count1207++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1208 = 0; count1208 < generation\_parameter\_z; count1208++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1209 = 0; count1209 < generation\_parameter\_x; count1209++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1210 = 0; count1210 < generation\_parameter\_y; count1210++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1211 = 0; count1211 < generation\_parameter\_z; count1211++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1212 = 0; count1212 < generation\_parameter\_x; count1212++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1213 = 0; count1213 < generation\_parameter\_y; count1213++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1214 = 0; count1214 < generation\_parameter\_z; count1214++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1215 = 0; count1215 < generation\_parameter\_x; count1215++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1216 = 0; count1216 < generation\_parameter\_y; count1216++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1217 = 0; count1217 < generation\_parameter\_z; count1217++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* (5 \* 1.5);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

}

wires();

}

function wires() {

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1218 = 0; count1218 < 39990; count1218++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1219 = 0; count1219 < 39990; count1219++) {

starter\_x = starter\_x - 1.546981838e+30;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1220 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1220 = 0; count1220 < repeat\_end1220; count1220++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.445e+30) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.445e+30) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1221 = 0; count1221 < generation\_parameter\_x; count1221++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1222 = 0; count1222 < generation\_parameter\_y; count1222++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1223 = 0; count1223 < generation\_parameter\_z; count1223++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1224 = 0; count1224 < generation\_parameter\_x; count1224++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1225 = 0; count1225 < generation\_parameter\_y; count1225++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1226 = 0; count1226 < generation\_parameter\_z; count1226++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1227 = 0; count1227 < generation\_parameter\_x; count1227++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1228 = 0; count1228 < generation\_parameter\_y; count1228++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1229 = 0; count1229 < generation\_parameter\_z; count1229++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1230 = 0; count1230 < generation\_parameter\_x; count1230++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1231 = 0; count1231 < generation\_parameter\_y; count1231++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1232 = 0; count1232 < generation\_parameter\_z; count1232++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1233 = 0; count1233 < 5000; count1233++) {

starter\_y = starter\_y - 1.23758547e+31;

for (var count1234 = 0; count1234 < 5000; count1234++) {

starter\_x = starter\_x - 1.23758547e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1235 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1235 = 0; count1235 < repeat\_end1235; count1235++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.2e+31) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1236 = 0; count1236 < generation\_parameter\_x; count1236++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1237 = 0; count1237 < generation\_parameter\_y; count1237++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1238 = 0; count1238 < generation\_parameter\_z; count1238++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1239 = 0; count1239 < generation\_parameter\_x; count1239++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1240 = 0; count1240 < generation\_parameter\_y; count1240++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1241 = 0; count1241 < generation\_parameter\_z; count1241++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1242 = 0; count1242 < generation\_parameter\_x; count1242++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1243 = 0; count1243 < generation\_parameter\_y; count1243++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1244 = 0; count1244 < generation\_parameter\_z; count1244++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1245 = 0; count1245 < generation\_parameter\_x; count1245++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1246 = 0; count1246 < generation\_parameter\_y; count1246++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1247 = 0; count1247 < generation\_parameter\_z; count1247++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 5000 \* 1.23758547e+31;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1248 = 0; count1248 < 1000; count1248++) {

starter\_y = starter\_y - 6.187927353e+31;

for (var count1249 = 0; count1249 < 1000; count1249++) {

starter\_x = starter\_x - 6.187927353e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1250 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1250 = 0; count1250 < repeat\_end1250; count1250++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1251 = 0; count1251 < generation\_parameter\_x; count1251++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1252 = 0; count1252 < generation\_parameter\_y; count1252++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1253 = 0; count1253 < generation\_parameter\_z; count1253++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1254 = 0; count1254 < generation\_parameter\_x; count1254++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1255 = 0; count1255 < generation\_parameter\_y; count1255++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1256 = 0; count1256 < generation\_parameter\_z; count1256++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1257 = 0; count1257 < generation\_parameter\_x; count1257++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1258 = 0; count1258 < generation\_parameter\_y; count1258++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1259 = 0; count1259 < generation\_parameter\_z; count1259++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1260 = 0; count1260 < generation\_parameter\_x; count1260++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1261 = 0; count1261 < generation\_parameter\_y; count1261++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1262 = 0; count1262 < generation\_parameter\_z; count1262++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x \* (1000 + 6.187927353e+31);

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1263 = 0; count1263 < 500; count1263++) {

starter\_y = starter\_y - 6.187927353e+31 \* 2;

for (var count1264 = 0; count1264 < 500; count1264++) {

starter\_x = starter\_x - 6.187927353e+31 \* 2;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1265 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1265 = 0; count1265 < repeat\_end1265; count1265++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 2) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 2) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1266 = 0; count1266 < generation\_parameter\_x; count1266++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1267 = 0; count1267 < generation\_parameter\_y; count1267++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1268 = 0; count1268 < generation\_parameter\_z; count1268++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1269 = 0; count1269 < generation\_parameter\_x; count1269++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1270 = 0; count1270 < generation\_parameter\_y; count1270++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1271 = 0; count1271 < generation\_parameter\_z; count1271++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1272 = 0; count1272 < generation\_parameter\_x; count1272++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1273 = 0; count1273 < generation\_parameter\_y; count1273++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1274 = 0; count1274 < generation\_parameter\_z; count1274++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1275 = 0; count1275 < generation\_parameter\_x; count1275++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1276 = 0; count1276 < generation\_parameter\_y; count1276++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1277 = 0; count1277 < generation\_parameter\_z; count1277++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1278 = 0; count1278 < 200; count1278++) {

starter\_y = starter\_y - 6.187927353e+31 \* 5;

for (var count1279 = 0; count1279 < 200; count1279++) {

starter\_x = starter\_x - 6.187927353e+31 \* 5;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1280 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1280 = 0; count1280 < repeat\_end1280; count1280++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 5) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 5) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1281 = 0; count1281 < generation\_parameter\_x; count1281++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1282 = 0; count1282 < generation\_parameter\_y; count1282++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1283 = 0; count1283 < generation\_parameter\_z; count1283++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1284 = 0; count1284 < generation\_parameter\_x; count1284++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1285 = 0; count1285 < generation\_parameter\_y; count1285++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1286 = 0; count1286 < generation\_parameter\_z; count1286++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1287 = 0; count1287 < generation\_parameter\_x; count1287++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1288 = 0; count1288 < generation\_parameter\_y; count1288++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1289 = 0; count1289 < generation\_parameter\_z; count1289++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1290 = 0; count1290 < generation\_parameter\_x; count1290++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1291 = 0; count1291 < generation\_parameter\_y; count1291++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1292 = 0; count1292 < generation\_parameter\_z; count1292++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1293 = 0; count1293 < 100; count1293++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

for (var count1294 = 0; count1294 < 100; count1294++) {

starter\_x = starter\_x - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1295 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1295 = 0; count1295 < repeat\_end1295; count1295++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 10) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1296 = 0; count1296 < generation\_parameter\_x; count1296++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1297 = 0; count1297 < generation\_parameter\_y; count1297++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1298 = 0; count1298 < generation\_parameter\_z; count1298++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1299 = 0; count1299 < generation\_parameter\_x; count1299++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1300 = 0; count1300 < generation\_parameter\_y; count1300++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1301 = 0; count1301 < generation\_parameter\_z; count1301++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1302 = 0; count1302 < generation\_parameter\_x; count1302++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1303 = 0; count1303 < generation\_parameter\_y; count1303++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1304 = 0; count1304 < generation\_parameter\_z; count1304++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1305 = 0; count1305 < generation\_parameter\_x; count1305++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1306 = 0; count1306 < generation\_parameter\_y; count1306++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1307 = 0; count1307 < generation\_parameter\_z; count1307++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

aluminium\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* (5 \* 1.5);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1308 = 0; count1308 < 39990; count1308++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1309 = 0; count1309 < 39990; count1309++) {

starter\_x = starter\_x - 1.546981838e+30;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1310 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1310 = 0; count1310 < repeat\_end1310; count1310++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.445e+30) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.445e+30) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.445e+30 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1311 = 0; count1311 < generation\_parameter\_x; count1311++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1312 = 0; count1312 < generation\_parameter\_y; count1312++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1313 = 0; count1313 < generation\_parameter\_z; count1313++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1314 = 0; count1314 < generation\_parameter\_x; count1314++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1315 = 0; count1315 < generation\_parameter\_y; count1315++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1316 = 0; count1316 < generation\_parameter\_z; count1316++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1317 = 0; count1317 < generation\_parameter\_x; count1317++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1318 = 0; count1318 < generation\_parameter\_y; count1318++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1319 = 0; count1319 < generation\_parameter\_z; count1319++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1320 = 0; count1320 < generation\_parameter\_x; count1320++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1321 = 0; count1321 < generation\_parameter\_y; count1321++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1322 = 0; count1322 < generation\_parameter\_z; count1322++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1323 = 0; count1323 < 5000; count1323++) {

starter\_y = starter\_y - 1.23758547e+31;

for (var count1324 = 0; count1324 < 5000; count1324++) {

starter\_x = starter\_x - 1.23758547e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1325 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1325 = 0; count1325 < repeat\_end1325; count1325++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 1.2e+31) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 1.2e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_y = Math.round(1.2e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1326 = 0; count1326 < generation\_parameter\_x; count1326++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1327 = 0; count1327 < generation\_parameter\_y; count1327++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1328 = 0; count1328 < generation\_parameter\_z; count1328++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1329 = 0; count1329 < generation\_parameter\_x; count1329++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1330 = 0; count1330 < generation\_parameter\_y; count1330++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1331 = 0; count1331 < generation\_parameter\_z; count1331++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1332 = 0; count1332 < generation\_parameter\_x; count1332++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1333 = 0; count1333 < generation\_parameter\_y; count1333++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1334 = 0; count1334 < generation\_parameter\_z; count1334++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1335 = 0; count1335 < generation\_parameter\_x; count1335++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1336 = 0; count1336 < generation\_parameter\_y; count1336++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1337 = 0; count1337 < generation\_parameter\_z; count1337++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 5000 \* 1.23758547e+31;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1338 = 0; count1338 < 1000; count1338++) {

starter\_y = starter\_y - 6.187927353e+31;

for (var count1339 = 0; count1339 < 1000; count1339++) {

starter\_x = starter\_x - 6.187927353e+31;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1340 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1340 = 0; count1340 < repeat\_end1340; count1340++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_y = Math.round(6.18e+31 / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1341 = 0; count1341 < generation\_parameter\_x; count1341++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1342 = 0; count1342 < generation\_parameter\_y; count1342++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1343 = 0; count1343 < generation\_parameter\_z; count1343++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1344 = 0; count1344 < generation\_parameter\_x; count1344++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1345 = 0; count1345 < generation\_parameter\_y; count1345++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1346 = 0; count1346 < generation\_parameter\_z; count1346++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1347 = 0; count1347 < generation\_parameter\_x; count1347++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1348 = 0; count1348 < generation\_parameter\_y; count1348++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1349 = 0; count1349 < generation\_parameter\_z; count1349++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1350 = 0; count1350 < generation\_parameter\_x; count1350++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1351 = 0; count1351 < generation\_parameter\_y; count1351++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1352 = 0; count1352 < generation\_parameter\_z; count1352++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x \* (1000 + 6.187927353e+31);

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1353 = 0; count1353 < 500; count1353++) {

starter\_y = starter\_y - 6.187927353e+31 \* 2;

for (var count1354 = 0; count1354 < 500; count1354++) {

starter\_x = starter\_x - 6.187927353e+31 \* 2;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1355 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1355 = 0; count1355 < repeat\_end1355; count1355++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 2) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 2) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 2) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1356 = 0; count1356 < generation\_parameter\_x; count1356++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1357 = 0; count1357 < generation\_parameter\_y; count1357++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1358 = 0; count1358 < generation\_parameter\_z; count1358++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1359 = 0; count1359 < generation\_parameter\_x; count1359++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1360 = 0; count1360 < generation\_parameter\_y; count1360++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1361 = 0; count1361 < generation\_parameter\_z; count1361++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1362 = 0; count1362 < generation\_parameter\_x; count1362++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1363 = 0; count1363 < generation\_parameter\_y; count1363++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1364 = 0; count1364 < generation\_parameter\_z; count1364++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1365 = 0; count1365 < generation\_parameter\_x; count1365++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1366 = 0; count1366 < generation\_parameter\_y; count1366++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1367 = 0; count1367 < generation\_parameter\_z; count1367++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1368 = 0; count1368 < 200; count1368++) {

starter\_y = starter\_y - 6.187927353e+31 \* 5;

for (var count1369 = 0; count1369 < 200; count1369++) {

starter\_x = starter\_x - 6.187927353e+31 \* 5;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1370 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1370 = 0; count1370 < repeat\_end1370; count1370++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 5) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 5) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 5) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1371 = 0; count1371 < generation\_parameter\_x; count1371++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1372 = 0; count1372 < generation\_parameter\_y; count1372++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1373 = 0; count1373 < generation\_parameter\_z; count1373++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1374 = 0; count1374 < generation\_parameter\_x; count1374++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1375 = 0; count1375 < generation\_parameter\_y; count1375++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1376 = 0; count1376 < generation\_parameter\_z; count1376++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1377 = 0; count1377 < generation\_parameter\_x; count1377++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1378 = 0; count1378 < generation\_parameter\_y; count1378++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1379 = 0; count1379 < generation\_parameter\_z; count1379++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1380 = 0; count1380 < generation\_parameter\_x; count1380++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1381 = 0; count1381 < generation\_parameter\_y; count1381++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1382 = 0; count1382 < generation\_parameter\_z; count1382++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1383 = 0; count1383 < 100; count1383++) {

starter\_y = starter\_y - 6.187927353e+31 \* 10;

for (var count1384 = 0; count1384 < 100; count1384++) {

starter\_x = starter\_x - 6.187927353e+31 \* 10;

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1385 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1385 = 0; count1385 < repeat\_end1385; count1385++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.18e+31 \* 10) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.18e+31 \* 10) {

if (string\_z < starter\_z + 6.187865474e+34) {

if (string\_z > starter\_z) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

used\_density\_parameter = 2.225797469e+25;

generation\_parameter\_x = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_y = Math.round((6.18e+31 \* 10) / used\_density\_parameter);

generation\_parameter\_z = Math.round(6.187927353e+34 / used\_density\_parameter);

for (var count1386 = 0; count1386 < generation\_parameter\_x; count1386++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1387 = 0; count1387 < generation\_parameter\_y; count1387++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1388 = 0; count1388 < generation\_parameter\_z; count1388++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1389 = 0; count1389 < generation\_parameter\_x; count1389++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1390 = 0; count1390 < generation\_parameter\_y; count1390++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1391 = 0; count1391 < generation\_parameter\_z; count1391++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1392 = 0; count1392 < generation\_parameter\_x; count1392++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1393 = 0; count1393 < generation\_parameter\_y; count1393++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1394 = 0; count1394 < generation\_parameter\_z; count1394++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1395 = 0; count1395 < generation\_parameter\_x; count1395++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1396 = 0; count1396 < generation\_parameter\_y; count1396++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1397 = 0; count1397 < generation\_parameter\_z; count1397++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

copper\_metallic\_();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 6.187927353e+31 \* 1000;

}

}

centerpoint\_z = centerpoint\_z - (3.093963676e+34 \* 2) \* (5 \* 1.5);

centerpoint\_x = centerpoint\_x - (3.093963676e+34 \* 2) \* (1 \* 1.5);

}

}

nails();

}

function nails() {

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1398 = 0; count1398 < 9; count1398++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1399 = 0; count1399 < 9; count1399++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round(6.187927353e+33 / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1400 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1400 = 0; count1400 < repeat\_end1400; count1400++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1401 = 0; count1401 < generation\_parameter\_x; count1401++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1402 = 0; count1402 < generation\_parameter\_y; count1402++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1403 = 0; count1403 < generation\_parameter\_z; count1403++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1404 = 0; count1404 < generation\_parameter\_x; count1404++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1405 = 0; count1405 < generation\_parameter\_y; count1405++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1406 = 0; count1406 < generation\_parameter\_z; count1406++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1407 = 0; count1407 < generation\_parameter\_x; count1407++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1408 = 0; count1408 < generation\_parameter\_y; count1408++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1409 = 0; count1409 < generation\_parameter\_z; count1409++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1410 = 0; count1410 < generation\_parameter\_x; count1410++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1411 = 0; count1411 < generation\_parameter\_y; count1411++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1412 = 0; count1412 < generation\_parameter\_z; count1412++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1413 = 0; count1413 < 18; count1413++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1414 = 0; count1414 < 18; count1414++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.187927353e+33 / 2) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1415 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1415 = 0; count1415 < repeat\_end1415; count1415++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1416 = 0; count1416 < generation\_parameter\_x; count1416++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1417 = 0; count1417 < generation\_parameter\_y; count1417++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1418 = 0; count1418 < generation\_parameter\_z; count1418++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1419 = 0; count1419 < generation\_parameter\_x; count1419++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1420 = 0; count1420 < generation\_parameter\_y; count1420++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1421 = 0; count1421 < generation\_parameter\_z; count1421++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1422 = 0; count1422 < generation\_parameter\_x; count1422++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1423 = 0; count1423 < generation\_parameter\_y; count1423++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1424 = 0; count1424 < generation\_parameter\_z; count1424++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1425 = 0; count1425 < generation\_parameter\_x; count1425++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1426 = 0; count1426 < generation\_parameter\_y; count1426++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1427 = 0; count1427 < generation\_parameter\_z; count1427++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1428 = 0; count1428 < 36; count1428++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1429 = 0; count1429 < 36; count1429++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.187927353e+33 / 4) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1430 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1430 = 0; count1430 < repeat\_end1430; count1430++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1431 = 0; count1431 < generation\_parameter\_x; count1431++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1432 = 0; count1432 < generation\_parameter\_y; count1432++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1433 = 0; count1433 < generation\_parameter\_z; count1433++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1434 = 0; count1434 < generation\_parameter\_x; count1434++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1435 = 0; count1435 < generation\_parameter\_y; count1435++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1436 = 0; count1436 < generation\_parameter\_z; count1436++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1437 = 0; count1437 < generation\_parameter\_x; count1437++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1438 = 0; count1438 < generation\_parameter\_y; count1438++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1439 = 0; count1439 < generation\_parameter\_z; count1439++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1440 = 0; count1440 < generation\_parameter\_x; count1440++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1441 = 0; count1441 < generation\_parameter\_y; count1441++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1442 = 0; count1442 < generation\_parameter\_z; count1442++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1443 = 0; count1443 < 72; count1443++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1444 = 0; count1444 < 72; count1444++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.187927353e+33 / 8) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1445 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1445 = 0; count1445 < repeat\_end1445; count1445++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1446 = 0; count1446 < generation\_parameter\_x; count1446++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1447 = 0; count1447 < generation\_parameter\_y; count1447++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1448 = 0; count1448 < generation\_parameter\_z; count1448++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1449 = 0; count1449 < generation\_parameter\_x; count1449++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1450 = 0; count1450 < generation\_parameter\_y; count1450++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1451 = 0; count1451 < generation\_parameter\_z; count1451++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1452 = 0; count1452 < generation\_parameter\_x; count1452++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1453 = 0; count1453 < generation\_parameter\_y; count1453++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1454 = 0; count1454 < generation\_parameter\_z; count1454++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1455 = 0; count1455 < generation\_parameter\_x; count1455++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1456 = 0; count1456 < generation\_parameter\_y; count1456++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1457 = 0; count1457 < generation\_parameter\_z; count1457++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1458 = 0; count1458 < 144; count1458++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1459 = 0; count1459 < 144; count1459++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.187927353e+33 / 16) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1460 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1460 = 0; count1460 < repeat\_end1460; count1460++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1461 = 0; count1461 < generation\_parameter\_x; count1461++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1462 = 0; count1462 < generation\_parameter\_y; count1462++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1463 = 0; count1463 < generation\_parameter\_z; count1463++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1464 = 0; count1464 < generation\_parameter\_x; count1464++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1465 = 0; count1465 < generation\_parameter\_y; count1465++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1466 = 0; count1466 < generation\_parameter\_z; count1466++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1467 = 0; count1467 < generation\_parameter\_x; count1467++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1468 = 0; count1468 < generation\_parameter\_y; count1468++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1469 = 0; count1469 < generation\_parameter\_z; count1469++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1470 = 0; count1470 < generation\_parameter\_x; count1470++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1471 = 0; count1471 < generation\_parameter\_y; count1471++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1472 = 0; count1472 < generation\_parameter\_z; count1472++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

if (unused\_variable == 5) {

package2();

if (unused\_variable == 5) {

starter\_y = centerpoint\_y - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_x = centerpoint\_x - (3.093963676e+32 \* 2 + 3.093963676e+29);

starter\_z = centerpoint\_z + (3.093963676e+32 \* 2 + 3.093963676e+29);

}

if (unused\_variable == 5) {

for (var count1473 = 0; count1473 < 288; count1473++) {

starter\_y = starter\_y - 1.546981838e+30;

for (var count1474 = 0; count1474 < 288; count1474++) {

starter\_x = starter\_x - 1.546981838e+30;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.round((6.187927353e+33 / 32) / used\_density\_parameter);

generation\_parameter\_y = Math.round(generation\_parameter\_x);

generation\_parameter\_z = Math.round(generation\_parameter\_x \* 10);

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1475 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1475 = 0; count1475 < repeat\_end1475; count1475++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < starter\_x) {

if (string\_x > starter\_x - used\_density\_parameter \* generation\_parameter\_x) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - used\_density\_parameter \* generation\_parameter\_y) {

if (string\_z < starter\_z + used\_density\_parameter \* generation\_parameter\_z) {

if (string\_z > starter\_z) {

if (string\_z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

} else if (Math.abs(starter\_y - string\_y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_x - string\_x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(string\_z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (unused\_variable == 5) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

for (var count1476 = 0; count1476 < generation\_parameter\_x; count1476++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1477 = 0; count1477 < generation\_parameter\_y; count1477++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1478 = 0; count1478 < generation\_parameter\_z; count1478++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

centerpoint\_y = cursor1y + used\_density\_parameter / 2;

for (var count1479 = 0; count1479 < generation\_parameter\_x; count1479++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1480 = 0; count1480 < generation\_parameter\_y; count1480++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1481 = 0; count1481 < generation\_parameter\_z; count1481++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1482 = 0; count1482 < generation\_parameter\_x; count1482++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1483 = 0; count1483 < generation\_parameter\_y; count1483++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1484 = 0; count1484 < generation\_parameter\_z; count1484++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = starter\_x;

cursor1y = starter\_y;

cursor1z = starter\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1x = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (var count1485 = 0; count1485 < generation\_parameter\_x; count1485++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1486 = 0; count1486 < generation\_parameter\_y; count1486++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1487 = 0; count1487 < generation\_parameter\_z; count1487++) {

cursor1z = cursor1z + used\_density\_parameter;

if (cursor1z < starter\_z + (used\_density\_parameter \* generation\_parameter\_z) / 100) {

set\_earth\_rotation();

aluminium\_metallic\_();

} else if (Math.abs(starter\_y - cursor1y) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

if (Math.abs(starter\_y - cursor1x) < ((used\_density\_parameter \* generation\_parameter\_y) \* (Math.abs(used\_density\_parameter \* generation\_parameter\_z - Math.abs(cursor1z - starter\_z)) / ((used\_density\_parameter \* generation\_parameter\_z) / 100))) / 2) {

set\_earth\_rotation();

aluminium\_metallic\_();

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

starter\_x = starter\_x + 39990 \* 1.546981838e+30;

}

}

centerpoint\_z = centerpoint\_z + (3.093963676e+34 \* 2) \* 1.5;

}

}

}

function prevolve() {

unused\_variable = 5;

}

//unfinished, for ancestral sequence reconstruction

function evolve() {

evolength = Math.round(3.06302404e+38 / (((sizestart + sizeend) / 2) \* 1.3 + 8.044305559e+33));

if (evar == 100) {

evolength = evolength \* 10;

}

}

function obtain\_evogen() {

gentarget = genomelist[evotaxon - 1];

get\_genetic\_text();

genderiv = '';

intextnum = 1;

while (intextnum <= genraw.length) {

if (genraw.charAt(intextnum - 1) == 'a' || genraw.charAt(intextnum - 1) == 'A') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'g' || genraw.charAt(intextnum - 1) == 'G') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'c' || genraw.charAt(intextnum - 1) == 'C') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if ((genraw.charAt(intextnum - 1) == 't' || genraw.charAt(intextnum - 1) == 'T') || (genraw.charAt(intextnum - 1) == 'u' || genraw.charAt(intextnum - 1) == 'U')) {

genderiv = String(genderiv) + String('T');

}

intextnum = intextnum + 1;

}

gena = genderiv;

if (evotaxon == genomelist.length || evotaxon == genomelist.length - 1) {

gentarget = genomelist[(evotaxon - 1) - 1];

} else {

gentarget = genomelist[(evotaxon + 1) - 1];

}

get\_genetic\_text();

genderiv = '';

intextnum = 1;

while (intextnum <= genraw.length) {

if (genraw.charAt(intextnum - 1) == 'a' || genraw.charAt(intextnum - 1) == 'A') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'g' || genraw.charAt(intextnum - 1) == 'G') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'c' || genraw.charAt(intextnum - 1) == 'C') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if ((genraw.charAt(intextnum - 1) == 't' || genraw.charAt(intextnum - 1) == 'T') || (genraw.charAt(intextnum - 1) == 'u' || genraw.charAt(intextnum - 1) == 'U')) {

genderiv = String(genderiv) + String('T');

}

intextnum = intextnum + 1;

}

genb = genderiv;

var recseq;

recseq = "CCCACCC";

var repeat\_end = Math.ceil((evoturn / evolength) \* 10000);

for (var count4 = 0; count4 < repeat\_end; count4++) {

recseq = String(recseq) + String('CCCACCC');

}

genoutput = gena + recseq + genb;

evogen = new Blob([genoutput]);

if ((evoturn - 1) / evolength >= 1) {

areached = true;

}

}

async function get\_genetic\_text() {

genraw = await gentarget.text();

}

function sugar\_cube() {

convert\_coordinates();

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1488 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1488 = 0; count1488 < repeat\_end1488; count1488++) {

if (string\_solar\_\_distance\_parameter + 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

if (string\_x < starter\_x) {

if (string\_x > starter\_x - 6.187865474e+33) {

if (string\_y < starter\_y) {

if (string\_y > starter\_y - 6.187865474e+33) {

if (string\_z < starter\_z + 6.187865474e+33) {

if (string\_z > starter\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 2.503635407e+22;

generation\_parameter\_3 = Math.round(6.187927353e+33 / used\_density\_parameter);

for (var count1489 = 0; count1489 < generation\_parameter\_3; count1489++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1490 = 0; count1490 < generation\_parameter\_3; count1490++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1491 = 0; count1491 < generation\_parameter\_3; count1491++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_2();

}

cursor1z = cursor1z - generation\_parameter\_3 \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

function marsupial\_female() {

vertebrate\_female();

}

function marsupial\_male() {

vertebrate\_male();

}

function terrestrial() {

unused\_variable = 5;

material\_temperature\_\_kelvins = 309.15;

rho = 3.942347699e+41;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9629803737) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - 0.00021052631 \* (Math.PI / 180);

genomelist = evolutionary\_genomes;

evar = 1;

evotaxon = 1;

prevolve();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

ploidy = 2;

areached = false;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

ploidy = 2;

areached = false;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 1.4;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

ploidy = 2;

areached = false;

sizestart = 6.187927353e+34 \* 1.4;

sizeend = 6.187927353e+34 \* 1.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.6;

sizeend = 6.187927353e+34 \* 1.4;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.4;

sizeend = 6.187927353e+34 \* 1.19;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

human\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.3;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.3;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.3;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.035;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.91;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.87;

sizeend = 6.187927353e+34 \* 0.8;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.75;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.75;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.6;

sizeend = 6.187927353e+34 \* 0.75;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.71;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.445;

sizeend = 6.187927353e+34 \* 0.71;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.31;

sizeend = 6.187927353e+34 \* 0.75;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 0.65;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.07;

sizeend = 6.187927353e+34 \* 0.23;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.214;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.51;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.367;

sizeend = 6.187927353e+34 \* 0.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 0.367;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.22;

sizeend = 6.187927353e+34 \* 0.3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.22;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.49;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.65;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

evolution\_2();

}

function evolution\_2() {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.6;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.97;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.98;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.3;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.36;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.43;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.47;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.35;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.21;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.15;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.32;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.21;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.2;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.425;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.155;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.4243;

sizeend = 6.187927353e+34 \* 0.245;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.27;

sizeend = 6.187927353e+34 \* 0.26;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

evolution\_3();

}

function evolution\_3() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.45;

sizeend = 6.187927353e+34 \* 0.26;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.425;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.125;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.725;

sizeend = 6.187927353e+34 \* 0.33;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.42;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 1.12;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 0.635;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.9;

sizeend = 6.187927353e+34 \* 0.635;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.999;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.99;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.9;

sizeend = 6.187927353e+34 \* 1.1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.95;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.96;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.03;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.3;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.578;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.1;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 0.85;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.157;

sizeend = 6.187927353e+34 \* 0.653;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 0.653;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.55;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.587;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.01357;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.75;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.65;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.56;

sizeend = 6.187927353e+34 \* 1.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.57;

sizeend = 6.187927353e+34 \* 1.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 0.45;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.45;

sizeend = 6.187927353e+34 \* 0.451;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.125;

sizeend = 6.187927353e+34 \* 0.45;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.0957;

sizeend = 6.187927353e+34 \* 0.45;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5357;

sizeend = 6.187927353e+34 \* 0.45;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

evolution\_4();

}

function evolution\_4() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.2;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 2.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.35;

sizeend = 6.187927353e+34 \* 1.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 0.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.324179;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.8;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.2;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.4;

sizeend = 6.187927353e+34 \* 1.09;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 2.2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.57;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.85;

sizeend = 6.187927353e+34 \* 1.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 1.6;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 2.99;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 1.67;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.67;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.6;

sizeend = 6.187927353e+34 \* 1.67;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.6;

sizeend = 6.187927353e+34 \* 1.67;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.2;

sizeend = 6.187927353e+34 \* 0.26;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.4;

sizeend = 6.187927353e+34 \* 0.26;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 0.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 0.71;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.31;

sizeend = 6.187927353e+34 \* 0.71;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 0.2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.4;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.27;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.27;

sizeend = 6.187927353e+34 \* 0.21;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.75;

sizeend = 6.187927353e+34 \* 0.21;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi / 180 \* Math.PI);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi / 180 \* Math.PI) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

if (evoturn / evolength < 230 / 320) {

sizemax = 0.3;

} else if (evoturn / evolength < 265 / 320) {

sizemax = 0.6;

} else if (evoturn / evolength < 280 / 320) {

sizemax = 1.1;

}

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

marsupial\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

evolution\_5();

}

function evolution\_5() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.3;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.99;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.7;

sizeend = 6.187927353e+34 \* 2.3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.7;

sizeend = 6.187927353e+34 \* 2.3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.7;

sizeend = 6.187927353e+34 \* 2.3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.7;

sizeend = 6.187927353e+34 \* 2.3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

if (evoturn / evolength > 60 / 320) {

if (evoturn / evolength < 140 / 320) {

sizemax = 6.187927353e+34 \* 0.9;

}

}

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 1.7;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.95;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.3;

sizeend = 6.187927353e+34 \* 0.8;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.15;

sizeend = 6.187927353e+34 \* 0.8;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

evolution\_6();

}

function evolution\_6() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.5;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.6;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.25;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.53475;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.9;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.2;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.7;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.2;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.8;

sizeend = 6.187927353e+34 \* 1.01;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 2.0000001;

widthmax = 10;

evolve();

while (areached == false) {

for (var count1492 = 0; count1492 < 10; count1492++) {

if (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

}

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - ((((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1) \* 20;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 2.0000001;

widthmax = 10;

evolve();

while (areached == false) {

for (var count1493 = 0; count1493 < 10; count1493++) {

if (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

}

cursor\_eta = cursor\_eta + ((((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1) \* 20;

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - ((((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1) \* 20;

}

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* (math\_random\_int(1e+44, 1e+46) / 1e+47);

sizeend = sizestart + 100000000000000000000;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

cursor\_phi = (90 - (20.8718435868 + math\_random\_int(1, 2.5e+43) / 1e+44)) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* (math\_random\_int(1e+44, 1e+46) / 1e+47);

sizeend = sizestart + 100000000000000000000;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

evar = 100;

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* (math\_random\_int(1e+46, 4e+47) / 1e+47);

sizeend = sizestart + 100000000000000000000;

widthmax = 10;

evolve();

while (areached == false) {

for (var count1494 = 0; count1494 < 10; count1494++) {

if (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

}

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

cursor\_eta = cursor\_eta + ((((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1) \* 20;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - ((((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1) \* 20;

}

evar = 1;

}

evolution\_7();

}

function evolution\_7() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.003;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 2;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.003;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 2;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.04;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.001;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 2;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.03;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.013;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.013;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.013;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.02;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.028;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.100001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.100001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.100001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.05;

sizeend = 0.100001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.100001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

}

evolution\_8();

}

function evolution\_8() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.03;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.03;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 1;

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 2;

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.04;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 1;

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 2;

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.01;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 1;

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

ploidy = 2;

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.01;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.004;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.01;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.01;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.06;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.03;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.015;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.005;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.015;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.015;

sizeend = 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

sugar\_cube();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* ((0.1 \* 6.187927353e+34) / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* ((6.187927353e+34 \* 0.2) / 6.187927353e+34)) \* 1;

}

}

placeholder\_eta = cursor\_eta;

}

function freshwater() {

unused\_variable = 5;

material\_temperature\_\_kelvins = 309.15;

rho = 3.942328517e+41;

cursor\_phi = (90 - 20.8728435868) \* (Math.PI / 180);

cursor\_eta = (180 + 86.951822479) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (2 / 95) \* (Math.PI / 180);

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

if (evoturn / evolength < 15 / 600) {

ploidy = 4;

} else {

ploidy = 2;

}

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 3.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 3.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 3.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

if (evoturn / evolength < 15 / 600) {

ploidy = 4;

} else {

ploidy = 2;

}

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 3.51;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 3.5;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 3.1;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1.7;

sizeend = 6.187927353e+34 \* 1.78;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

cursor\_eta = cursor\_eta - (2 / 95) \* (Math.PI / 180);

freshwater\_2();

}

function freshwater\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.1000001;

widthmax = 10;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8718435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (widthmax / 6.187927353e+34)) \* 1;

}

}

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1) \* 1) \* 1;

evar = 100;

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = math\_random\_int(6.187927353e+34 \* 2, 6.187927353e+34 \* 0.001);

sizeend = sizestart + 1e+21;

widthmax = 10;

evolve();

while (areached == false) {

for (var count1495 = 0; count1495 < 50; count1495++) {

if (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

}

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 10) \* (widthmax / 6.187927353e+34)) \* 100;

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8818435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 10) \* (widthmax / 6.187927353e+34)) \* 100;

}

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = math\_random\_int(6.187927353e+34 \* 0.1, 6.187927353e+34 \* 30);

sizeend = sizestart + 1e+21;

widthmax = 10;

evolve();

while (areached == false) {

for (var count1496 = 0; count1496 < 50; count1496++) {

if (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemid / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

}

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 10) \* (widthmax / 6.187927353e+34)) \* 100;

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 20.8818435868) \* (Math.PI / 180);

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 10) \* (widthmax / 6.187927353e+34)) \* 1;

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 10) \* (widthmax / 6.187927353e+34)) \* 1;

}

evar = 1;

}

function marine() {

unused\_variable = 5;

material\_temperature\_\_kelvins = 309.15;

rho = 3.942328517e+41;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

cursor\_eta = (180 + 86.9636119526) \* (Math.PI / 180);

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 3.1;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 5.5;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 5.5;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3;

sizeend = 6.187927353e+34 \* 3.1;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

cursor\_eta = cursor\_eta + (50 / 95) \* (Math.PI / 180);

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 22;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 22;

sizeend = 6.187927353e+34 \* 20;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 22;

sizeend = 6.187927353e+34 \* 20;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 22;

sizeend = 6.187927353e+34 \* 20;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 22;

sizeend = 6.187927353e+34 \* 20;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 10;

sizeend = 6.187927353e+34 \* 9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 9;

sizeend = 6.187927353e+34 \* 8;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 9;

sizeend = 6.187927353e+34 \* 8;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 9;

sizeend = 6.187927353e+34 \* 8;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 3.5;

sizeend = 6.187927353e+34 \* 2;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2.5;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 5;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 5;

sizeend = 6.187927353e+34 \* 3;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 10;

sizeend = 6.187927353e+34 \* 5;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

cursor\_eta = cursor\_eta + (20 / 95) \* (Math.PI / 180);

marine\_2();

}

function marine\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 2;

sizeend = 6.187927353e+34 \* 1.9;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 1;

sizeend = 6.187927353e+34 \* 0.9999999;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

cursor\_eta = cursor\_eta + (20 / 95) \* (Math.PI / 180);

marine\_3();

}

function marine\_3() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

marine\_4();

}

function marine\_4() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 0.4;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.5;

sizeend = 6.187927353e+34 \* 0.4;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.1;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = 6.187927353e+34 \* 0.35;

sizeend = 6.187927353e+34 \* 0.10001;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

}

}

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

evar = 100;

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = math\_random\_int(6.187927353e+34 \* 0.1, 6.187927353e+34 \* 30);

sizeend = sizestart + 1e+24;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 100;

evolve();

while (areached == false) {

for (var count1497 = 0; count1497 < 50; count1497++) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_male();

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

vertebrate\_female();

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 100;

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 200;

}

evolutionary\_genomes[evotaxon - 1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

sizestart = math\_random\_int(6.187927353e+34 \* 0.001, 6.187927353e+34 \* 10);

sizeend = sizestart + 1e+24;

widthmax = 10;

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 100;

evolve();

while (areached == false) {

for (var count1498 = 0; count1498 < 50; count1498++) {

obtain\_evogen();

genome = evogen;

sizemax = sizestart + (evoturn / evolength) \* (sizeend - sizestart);

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_male();

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

invertebrate\_female();

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1;

evoturn = evoturn + 1;

if (widthmax < sizemid) {

widthmax = sizemid;

}

}

cursor\_eta = cursor\_eta - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 100;

cursor\_phi = cursor\_phi - (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 1.3;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.09) \* (Math.PI / 180);

cursor\_eta = cursor\_eta + (((0.01745329251 / 95) / 1000) \* (sizemax / 6.187927353e+34)) \* 200;

}

evar = 1;

symbiogenesis();

unicellular();

}

function symbiogenesis() {

cursor\_eta = placeholder\_eta - ((1 / 95) / 100000) \* (Math.PI / 180);

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

sizestart = 6.187927353e+32;

sizeend = 6.187927353e+32;

if (unused\_variable == 5) {

evotaxon = 1;

evar = 2;

genomelist = symvolgen;

var repeat\_end1499 = symvolgen.length;

for (var count1499 = 0; count1499 < repeat\_end1499; count1499++) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

prokaryote\_package();

cursor\_phi = cursor\_phi - (((3 / 95) / 10000000) \* (Math.PI / 180)) \* 2;

evoturn = evoturn + 1;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

}

cursor\_eta = cursor\_eta - (((1 / 95) / 10000000) \* (Math.PI / 180)) \* 1;

}

}

}

function unicellular() {

cursor\_eta = cursor\_eta - ((1 / 95) / 100000) \* (Math.PI / 180);

eukaryotic\_genomes[0] = prokaryotic\_genomes[math\_random\_int(1, prokaryotic\_genomes.length) - 1];

eukaryotic\_genomes[1] = prokaryotic\_genomes[math\_random\_int(1, prokaryotic\_genomes.length) - 1];

plant\_genomes[0] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

plant\_genomes[1] = eukaryotic\_genomes[math\_random\_int(1, eukaryotic\_genomes.length) - 1];

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

sizestart = 6.187927353e+32;

sizeend = 6.187927353e+32;

if (unused\_variable == 5) {

evotaxon = 1;

evar = 2;

genomelist = plant\_genomes;

var repeat\_end1500 = plant\_genomes.length;

for (var count1500 = 0; count1500 < repeat\_end1500; count1500++) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

plant\_package();

cursor\_phi = cursor\_phi - (((3 / 95) / 10000000) \* (Math.PI / 180)) \* 2;

evoturn = evoturn + 1;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

}

cursor\_eta = cursor\_eta - (((1 / 95) / 10000000) \* (Math.PI / 180)) \* 1;

}

}

if (unused\_variable == 5) {

evotaxon = 1;

evar = 3;

genomelist = eukaryotic\_genomes;

var repeat\_end1501 = eukaryotic\_genomes.length;

for (var count1501 = 0; count1501 < repeat\_end1501; count1501++) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

fungus\_package();

cursor\_phi = cursor\_phi - (((3 / 95) / 10000000) \* (Math.PI / 180)) \* 2;

evoturn = evoturn + 1;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

}

cursor\_eta = cursor\_eta - (((1 / 95) / 10000000) \* (Math.PI / 180)) \* 1;

}

}

if (unused\_variable == 5) {

evotaxon = 1;

evar = 4;

genomelist = prokaryotic\_genomes;

var repeat\_end1502 = prokaryotic\_genomes.length;

for (var count1502 = 0; count1502 < repeat\_end1502; count1502++) {

if (unused\_variable == 5) {

evoturn = 1;

areached = false;

ploidy = 2;

evolve();

while (areached == false) {

obtain\_evogen();

genome = evogen;

centerpoint\_x = rho \* (Math.sin(cursor\_phi) \* Math.cos(cursor\_eta));

centerpoint\_y = rho \* (Math.sin(cursor\_phi) \* Math.sin(cursor\_eta));

centerpoint\_z = rho \* Math.cos(cursor\_phi);

prokaryote\_package();

cursor\_phi = cursor\_phi - (((3 / 95) / 10000000) \* (Math.PI / 180)) \* 2;

evoturn = evoturn + 1;

}

evotaxon = evotaxon + 1;

cursor\_phi = (90 - 21.257) \* (Math.PI / 180);

}

cursor\_eta = cursor\_eta - (((1 / 95) / 10000000) \* (Math.PI / 180)) \* 1;

}

}

}

function prokaryote\_package() {

convert\_coordinates();

set\_organism\_rotation();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1503 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1503 = 0; count1503 < repeat\_end1503; count1503++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+30) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 1.23758547e+31) {

if (string\_z < centerpoint\_z + 1.856378206e+31) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1504 = 0; count1504 < generation\_parameter\_x; count1504++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1505 = 0; count1505 < generation\_parameter\_3; count1505++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1506 = 0; count1506 < generation\_parameter\_x; count1506++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1507 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1507 = 0; count1507 < repeat\_end1507; count1507++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1508 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1508 = 0; count1508 < repeat\_end1508; count1508++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1509 = Math.round(6.187927353e+30 / (6.187927353e+29 \* 2) - 2);

for (var count1509 = 0; count1509 < repeat\_end1509; count1509++) {

spawnerz = spawnerz + 6.187927353e+29 \* 2;

set\_earth\_rotation();

supported\_cell();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round((6.187927353e+30 \* 2) / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

spawnerz = spawnerz + 6.187927353e+29 / 1;

var repeat\_end1510 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1510 = 0; count1510 < repeat\_end1510; count1510++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1511 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1511 = 0; count1511 < repeat\_end1511; count1511++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1512 = Math.round(6.187927353e+30 / (6.187927353e+29 \* 2) - 2);

for (var count1512 = 0; count1512 < repeat\_end1512; count1512++) {

spawnerz = spawnerz + 6.187927353e+29 \* 2;

set\_earth\_rotation();

supporting\_cell();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round((6.187927353e+30 \* 2) / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+30 \* 1.9;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1513 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1513 = 0; count1513 < repeat\_end1513; count1513++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1514 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1514 = 0; count1514 < repeat\_end1514; count1514++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

melanocyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 2;

}

used\_density\_parameter = 6e+23;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1515 = 0; count1515 < generation\_parameter\_x; count1515++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1516 = 0; count1516 < generation\_parameter\_3; count1516++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1517 = 0; count1517 < generation\_parameter\_x; count1517++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function fungus\_package() {

convert\_coordinates();

set\_organism\_rotation();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1518 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1518 = 0; count1518 < repeat\_end1518; count1518++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+30) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 1.23758547e+31) {

if (string\_z < centerpoint\_z + 1.856378206e+31) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1519 = 0; count1519 < generation\_parameter\_x; count1519++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1520 = 0; count1520 < generation\_parameter\_3; count1520++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1521 = 0; count1521 < generation\_parameter\_x; count1521++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1522 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1522 = 0; count1522 < repeat\_end1522; count1522++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1523 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1523 = 0; count1523 < repeat\_end1523; count1523++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1524 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1524 = 0; count1524 < repeat\_end1524; count1524++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

lichen\_symbiont\_genome2();

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

spawnery = centerpoint\_y - 6.187927353e+30 / 1;

var repeat\_end1525 = Math.round((6.187927353e+30 / 2) / 6.187927353e+29 - 2);

for (var count1525 = 0; count1525 < repeat\_end1525; count1525++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1526 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1526 = 0; count1526 < repeat\_end1526; count1526++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1527 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1527 = 0; count1527 < repeat\_end1527; count1527++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

lichen\_cyanobiont\_genome2();

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

spawnerx = centerpoint\_x - 6.187927353e+30 / 2;

spawnery = centerpoint\_y - 6.187927353e+30 / 1;

var repeat\_end1528 = Math.round((6.187927353e+30 / 2) / 6.187927353e+29 - 2);

for (var count1528 = 0; count1528 < repeat\_end1528; count1528++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1529 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1529 = 0; count1529 < repeat\_end1529; count1529++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1530 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1530 = 0; count1530 < repeat\_end1530; count1530++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+30 \* 1.9;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1531 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1531 = 0; count1531 < repeat\_end1531; count1531++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1532 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1532 = 0; count1532 < repeat\_end1532; count1532++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

melanocyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 2;

}

used\_density\_parameter = 6e+23;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1533 = 0; count1533 < generation\_parameter\_x; count1533++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1534 = 0; count1534 < generation\_parameter\_3; count1534++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1535 = 0; count1535 < generation\_parameter\_x; count1535++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function plant\_package() {

convert\_coordinates();

set\_organism\_rotation();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1536 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1536 = 0; count1536 < repeat\_end1536; count1536++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - 6.187927353e+30) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - 1.23758547e+31) {

if (string\_z < centerpoint\_z + 1.856378206e+31) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1537 = 0; count1537 < generation\_parameter\_x; count1537++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1538 = 0; count1538 < generation\_parameter\_3; count1538++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1539 = 0; count1539 < generation\_parameter\_x; count1539++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_freshwater\_molecule();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = 6.187927353e+30 + centerpoint\_z;

}

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 120);

generation\_parameter\_x = Math.round((6.187927353e+30 / 2) / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 120);

for (var count1540 = 0; count1540 < generation\_parameter\_x; count1540++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1541 = 0; count1541 < generation\_parameter\_3; count1541++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1542 = 0; count1542 < generation\_parameter\_z; count1542++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = 6.187927353e+30 + centerpoint\_z;

}

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round((6.187927353e+30 / 2) / used\_density\_parameter - 60);

for (var count1543 = 0; count1543 < generation\_parameter\_x; count1543++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1544 = 0; count1544 < generation\_parameter\_3; count1544++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1545 = 0; count1545 < generation\_parameter\_z; count1545++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1x = centerpoint\_x - 3.093963676e+30;

cursor1z = centerpoint\_z + 6.187927353e+30;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 120);

generation\_parameter\_x = Math.round((6.187927353e+30 / 2) / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 120);

for (var count1546 = 0; count1546 < generation\_parameter\_x; count1546++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1547 = 0; count1547 < generation\_parameter\_3; count1547++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1548 = 0; count1548 < generation\_parameter\_z; count1548++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1z = centerpoint\_z + 6.187927353e+30;

cursor1x = centerpoint\_x - 6.187927353e+30 / 2;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round((6.187927353e+30 / 2) / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1549 = 0; count1549 < generation\_parameter\_x; count1549++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1550 = 0; count1550 < generation\_parameter\_3; count1550++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1551 = 0; count1551 < generation\_parameter\_z; count1551++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

root\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+30 / 2.5;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1552 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1552 = 0; count1552 < repeat\_end1552; count1552++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1553 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1553 = 0; count1553 < repeat\_end1553; count1553++) {

spawnery = spawnery - 6.187927353e+29;

var repeat\_end1554 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1554 = 0; count1554 < repeat\_end1554; count1554++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

omnicyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

spawnery = spawnery + 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 6.187927353e+30 \* 1.4;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 1;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_3 = Math.round((3.093963676e+30 / 1.25) / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 120);

generation\_parameter\_x = Math.round(3.093963676e+30 / used\_density\_parameter - 120);

for (var count1555 = 0; count1555 < generation\_parameter\_x; count1555++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1556 = 0; count1556 < generation\_parameter\_3; count1556++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1557 = 0; count1557 < generation\_parameter\_z; count1557++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 6.187927353e+30 \* 1.4;

centerpoint\_z = centerpoint\_z + 6.187927353e+30 \* 1;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(3.093963676e+30 / used\_density\_parameter - 60);

for (var count1558 = 0; count1558 < generation\_parameter\_x; count1558++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1559 = 0; count1559 < generation\_parameter\_3; count1559++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1560 = 0; count1560 < generation\_parameter\_z; count1560++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

callus\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 6.187927353e+30 \* 1.4;

cursor1x = centerpoint\_x - 6.187927353e+30 / 2;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 1;

used\_density\_parameter = 3.093963676e+26;

generation\_parameter\_x = Math.round(3.093963676e+30 / used\_density\_parameter - 120);

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 120);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 120);

for (var count1561 = 0; count1561 < generation\_parameter\_3; count1561++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1562 = 0; count1562 < generation\_parameter\_3; count1562++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1563 = 0; count1563 < generation\_parameter\_z; count1563++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = centerpoint\_y - 6.187927353e+30 \* 1.4;

cursor1x = centerpoint\_x - 6.187927353e+30 / 2;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 1;

used\_density\_parameter = 1.922068735e+25;

generation\_parameter\_x = Math.round(3.093963676e+30 / used\_density\_parameter - 60);

generation\_parameter\_3 = Math.round((6.187927353e+30 / 2.5) / used\_density\_parameter - 60);

generation\_parameter\_z = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1564 = 0; count1564 < generation\_parameter\_x; count1564++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1565 = 0; count1565 < generation\_parameter\_3; count1565++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1566 = 0; count1566 < generation\_parameter\_z; count1566++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

shoot\_medium();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y - 6.187927353e+30 \* 1.9;

spawnerz = centerpoint\_z;

}

spawnerz = centerpoint\_z + 6.187927353e+30 / 1;

var repeat\_end1567 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1567 = 0; count1567 < repeat\_end1567; count1567++) {

spawnerx = spawnerx - 6.187927353e+29;

var repeat\_end1568 = Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

for (var count1568 = 0; count1568 < repeat\_end1568; count1568++) {

spawnerz = spawnerz + 6.187927353e+29;

set\_earth\_rotation();

melanocyte();

}

spawnerz = spawnerz - 6.187927353e+29 \* Math.round(6.187927353e+30 / 6.187927353e+29 - 2);

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z + 6.187927353e+30 \* 2;

}

used\_density\_parameter = 6e+23;

generation\_parameter\_3 = Math.round(1.856378206e+31 / used\_density\_parameter - 60);

generation\_parameter\_x = Math.round(6.187927353e+30 / used\_density\_parameter - 60);

for (var count1569 = 0; count1569 < generation\_parameter\_x; count1569++) {

cursor1x = cursor1x - used\_density\_parameter;

for (var count1570 = 0; count1570 < generation\_parameter\_3; count1570++) {

cursor1y = cursor1y - used\_density\_parameter;

for (var count1571 = 0; count1571 < generation\_parameter\_x; count1571++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

randomize\_nutrient\_3();

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_3 \* used\_density\_parameter;

}

}

}

}

function find2() {

num1 = 1;

var repeat\_end1572 = list\_tissues.length;

for (var count1572 = 0; count1572 < repeat\_end1572; count1572++) {

num2 = 1;

yesno = false;

var repeat\_end1573 = target1.length;

for (var count1573 = 0; count1573 < repeat\_end1573; count1573++) {

if (target1[num2 - 1] == (list\_tissues[num1 - 1])[0]) {

yesno = true;

}

num2 = num2 + 1;

}

if (yesno == true) {

if ((list\_tissues[num1 - 1])[4] > (list\_tissues[num1 - 1])[1]) {

if ((list\_tissues[num1 - 1])[5] < (list\_tissues[num1 - 1])[2]) {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc1[2]) {

find\_2();

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

find\_2();

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc1[2]) {

find\_2();

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

find\_2();

}

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[5] < (list\_tissues[num1 - 1])[2]) {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc1[2]) {

find\_2();

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

find\_2();

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc1[2]) {

find\_2();

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

find\_2();

}

}

}

}

}

}

}

}

}

}

num1 = num1 + 1;

}

}

function find\_2() {

if ((list\_tissues[num1 - 1])[4] > (list\_tissues[num1 - 1])[1]) {

if ((list\_tissues[num1 - 1])[5] < (list\_tissues[num1 - 1])[2]) {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = true;

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc2[2]) {

make = true;

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = true;

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc2[2]) {

make = true;

}

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[5] < (list\_tissues[num1 - 1])[2]) {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = true;

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc2[2]) {

make = true;

}

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[6] < (list\_tissues[num1 - 1])[3]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = true;

}

}

}

}

}

}

} else {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[1] >= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[2] <= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

if ((list\_tissues[num1 - 1])[3] <= loc2[2]) {

make = true;

}

}

}

}

}

}

}

}

}

}

function box3() {

if (box[3] >= loc1[0]) {

if (box[0] <= loc1[0]) {

if (box[4] <= loc1[1]) {

if (box[1] >= loc1[1]) {

if (box[5] <= loc1[2]) {

if (box[2] >= loc1[2]) {

if (box[3] >= loc2[0]) {

if (box[0] <= loc2[0]) {

if (box[4] <= loc2[1]) {

if (box[1] >= loc2[1]) {

if (box[5] <= loc2[2]) {

if (box[2] >= loc2[2]) {

make = true;

}

}

}

}

}

}

}

}

}

}

}

}

}

function layer() {

num1 = 1;

var repeat\_end1574 = list\_tissues.length;

for (var count1574 = 0; count1574 < repeat\_end1574; count1574++) {

num2 = 1;

yesno = false;

var repeat\_end1575 = target1.length;

for (var count1575 = 0; count1575 < repeat\_end1575; count1575++) {

if (target1[num2 - 1] == (list\_tissues[num1 - 1])[0]) {

yesno = true;

}

num2 = num2 + 1;

}

if (thickabs != null) {

if (yesno == true) {

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) < thickabs) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickabs) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickabs) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_2();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) < thickabs) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickabs) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickabs) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_2();

}

}

}

}

num1 = num1 + 1;

}

if (thickbod != null) {

if (yesno == true) {

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) < thickbod \* sizemax) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickbod \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickbod \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_2();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) < thickbod \* sizemax) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickbod \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickbod \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_2();

}

}

}

}

num1 = num1 + 1;

}

if (thickrel != null) {

if (yesno == true) {

if (Math.abs((list\_tissues[num1 - 1])[1] - (list\_tissues[num1 - 1])[4]) <= Math.abs((list\_tissues[num1 - 1])[2] - (list\_tissues[num1 - 1])[5]) && Math.abs((list\_tissues[num1 - 1])[1] - (list\_tissues[num1 - 1])[4]) <= Math.abs((list\_tissues[num1 - 1])[3] - (list\_tissues[num1 - 1])[6])) {

stwidth = Math.abs((list\_tissues[num1 - 1])[1] - (list\_tissues[num1 - 1])[4]);

} else if (Math.abs((list\_tissues[num1 - 1])[2] - (list\_tissues[num1 - 1])[5]) <= Math.abs((list\_tissues[num1 - 1])[1] - (list\_tissues[num1 - 1])[4]) && Math.abs((list\_tissues[num1 - 1])[2] - (list\_tissues[num1 - 1])[5]) <= Math.abs((list\_tissues[num1 - 1])[3] - (list\_tissues[num1 - 1])[6])) {

stwidth = Math.abs((list\_tissues[num1 - 1])[2] - (list\_tissues[num1 - 1])[5]);

} else {

stwidth = Math.abs((list\_tissues[num1 - 1])[3] - (list\_tissues[num1 - 1])[6]);

}

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) < thickrel \* stwidth) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickrel \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickrel \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_2();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) < thickrel \* stwidth) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) < thickrel \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_2();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) < thickrel \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_2();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_2();

}

}

}

}

num1 = num1 + 1;

}

}

}

function layer\_2() {

if (thickabs != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) < thickabs) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickabs) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickabs) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_3();

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) < thickabs) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickabs) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickabs) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickabs) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickabs) {

layer\_3();

}

}

}

}

}

if (thickbod != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) < thickbod \* sizemax) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickbod \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickbod \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_3();

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) < thickbod \* sizemax) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickbod \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickbod \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickbod \* sizemax) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickbod \* sizemax) {

layer\_3();

}

}

}

}

}

if (thickrel != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) < thickrel \* stwidth) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickrel \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickrel \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_3();

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) < thickrel \* stwidth) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) < thickrel \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_3();

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) < thickrel \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) < thickrel \* stwidth) {

layer\_3();

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) < thickrel \* stwidth) {

layer\_3();

}

}

}

}

}

}

function layer\_3() {

if (thickabs != null) {

if (yesno == true) {

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) > thickabs2) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickabs2) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickabs2) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

layer\_4();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) > thickabs2) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickabs2) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickabs2) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

layer\_4();

}

}

}

}

}

if (thickbod != null) {

if (yesno == true) {

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

layer\_4();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickbod2 \* sizemax) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

layer\_4();

}

}

}

}

}

if (thickrel != null) {

if (yesno == true) {

if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[1]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

layer\_4();

}

}

} else if (Math.abs(loc1[0] - (list\_tissues[num1 - 1])[4]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[2]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

layer\_4();

}

} else if (Math.abs(loc1[1] - (list\_tissues[num1 - 1])[5]) > thickrel2 \* stwidth) {

if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

layer\_4();

} else if (Math.abs(loc1[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

layer\_4();

}

}

}

}

}

}

function layer\_4() {

if (thickabs != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) > thickabs2) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickabs2) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickabs2) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

make = true;

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) > thickabs2) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickabs2) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickabs2) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickabs2) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickabs2) {

make = true;

}

}

}

}

}

if (thickbod != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

make = true;

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickbod2 \* sizemax) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickbod2 \* sizemax) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickbod2 \* sizemax) {

make = true;

}

}

}

}

}

if (thickrel != null) {

if (yesno == true) {

if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[1]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

make = true;

}

}

} else if (Math.abs(loc2[0] - (list\_tissues[num1 - 1])[4]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[2]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

make = true;

}

} else if (Math.abs(loc2[1] - (list\_tissues[num1 - 1])[5]) > thickrel2 \* stwidth) {

if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[3]) > thickrel2 \* stwidth) {

make = true;

} else if (Math.abs(loc2[2] - (list\_tissues[num1 - 1])[6]) > thickrel2 \* stwidth) {

make = true;

}

}

}

}

}

}

function screen\_tissues() {

num1 = 1;

var repeat\_end1576 = list\_tissues.length;

for (var count1576 = 0; count1576 < repeat\_end1576; count1576++) {

num2 = 1;

yesno = false;

var repeat\_end1577 = exempt.length;

for (var count1577 = 0; count1577 < repeat\_end1577; count1577++) {

if (exempt[num2 - 1] == (list\_tissues[num1 - 1])[0]) {

yesno = true;

}

num2 = num2 + 1;

}

if (yesno == false) {

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[1] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[1] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[2] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[2] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[6] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_tissues[num1 - 1])[4] >= loc1[0]) {

if ((list\_tissues[num1 - 1])[4] <= loc2[0]) {

if ((list\_tissues[num1 - 1])[5] <= loc1[1]) {

if ((list\_tissues[num1 - 1])[5] >= loc2[1]) {

if ((list\_tissues[num1 - 1])[3] <= loc1[2]) {

if ((list\_tissues[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if (loc1[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc1[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc1[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc1[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc1[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc1[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc2[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc2[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc2[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc2[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc2[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc2[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc1[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc1[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc1[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc1[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc1[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc2[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc2[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc1[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc1[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc1[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc1[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc1[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc2[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc2[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc1[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc2[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc2[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc2[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc2[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc2[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc1[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc1[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc2[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc2[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_tissues[num1 - 1])[1]) {

if (loc2[0] <= (list\_tissues[num1 - 1])[4]) {

if (loc2[1] <= (list\_tissues[num1 - 1])[2]) {

if (loc2[1] >= (list\_tissues[num1 - 1])[5]) {

if (loc1[2] <= (list\_tissues[num1 - 1])[3]) {

if (loc1[2] >= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if ((((list\_tissues[num1 - 1])[2] >= loc2[1] && (list\_tissues[num1 - 1])[2] <= loc1[1]) && ((list\_tissues[num1 - 1])[3] >= loc2[2] && (list\_tissues[num1 - 1])[3] <= loc1[2]) || ((list\_tissues[num1 - 1])[5] >= loc2[1] && (list\_tissues[num1 - 1])[5] <= loc1[1]) && ((list\_tissues[num1 - 1])[3] >= loc2[2] && (list\_tissues[num1 - 1])[3] <= loc1[2])) || (((list\_tissues[num1 - 1])[3] >= loc2[1] && (list\_tissues[num1 - 1])[3] <= loc1[1]) && ((list\_tissues[num1 - 1])[6] >= loc2[2] && (list\_tissues[num1 - 1])[6] <= loc1[2]) || ((list\_tissues[num1 - 1])[5] >= loc2[1] && (list\_tissues[num1 - 1])[5] <= loc1[1]) && ((list\_tissues[num1 - 1])[6] >= loc2[2] && (list\_tissues[num1 - 1])[6] <= loc1[2]))) {

if ((list\_tissues[num1 - 1])[1] <= loc1[0]) {

if ((list\_tissues[num1 - 1])[4] >= loc2[0]) {

make = false;

}

}

}

if ((((list\_tissues[num1 - 1])[3] <= loc1[2] && (list\_tissues[num1 - 1])[3] >= loc2[2]) && ((list\_tissues[num1 - 1])[1] >= loc1[0] && (list\_tissues[num1 - 1])[1] <= loc2[0]) || ((list\_tissues[num1 - 1])[4] >= loc1[0] && (list\_tissues[num1 - 1])[4] <= loc2[0]) && ((list\_tissues[num1 - 1])[3] <= loc1[2] && (list\_tissues[num1 - 1])[3] >= loc2[2])) || (((list\_tissues[num1 - 1])[1] >= loc1[0] && (list\_tissues[num1 - 1])[1] <= loc2[0]) && ((list\_tissues[num1 - 1])[6] <= loc1[2] && (list\_tissues[num1 - 1])[6] >= loc2[2]) || ((list\_tissues[num1 - 1])[4] >= loc1[0] && (list\_tissues[num1 - 1])[4] <= loc2[0]) && ((list\_tissues[num1 - 1])[6] <= loc1[2] && (list\_tissues[num1 - 1])[6] >= loc2[2]))) {

if ((list\_tissues[num1 - 1])[2] >= loc1[1]) {

if ((list\_tissues[num1 - 1])[5] <= loc2[1]) {

make = false;

}

}

}

if ((((list\_tissues[num1 - 1])[2] <= loc1[1] && (list\_tissues[num1 - 1])[2] >= loc2[1]) && ((list\_tissues[num1 - 1])[1] >= loc1[0] && (list\_tissues[num1 - 1])[1] <= loc2[0]) || ((list\_tissues[num1 - 1])[2] <= loc1[1] && (list\_tissues[num1 - 1])[2] >= loc2[1]) && ((list\_tissues[num1 - 1])[4] >= loc1[0] && (list\_tissues[num1 - 1])[4] <= loc2[0])) || (((list\_tissues[num1 - 1])[5] <= loc1[1] && (list\_tissues[num1 - 1])[5] >= loc2[1]) && ((list\_tissues[num1 - 1])[4] >= loc1[0] && (list\_tissues[num1 - 1])[4] <= loc2[0]) || ((list\_tissues[num1 - 1])[5] <= loc1[1] && (list\_tissues[num1 - 1])[5] >= loc2[1]) && ((list\_tissues[num1 - 1])[1] >= loc1[0] && (list\_tissues[num1 - 1])[1] <= loc2[0]))) {

if ((list\_tissues[num1 - 1])[3] >= loc1[2]) {

if ((list\_tissues[num1 - 1])[6] <= loc2[2]) {

make = false;

}

}

}

if (((loc1[1] >= (list\_tissues[num1 - 1])[5] && loc1[1] <= (list\_tissues[num1 - 1])[2]) && (loc2[2] >= (list\_tissues[num1 - 1])[6] && loc2[2] <= (list\_tissues[num1 - 1])[3]) || (loc2[1] >= (list\_tissues[num1 - 1])[5] && loc2[1] <= (list\_tissues[num1 - 1])[2]) && (loc2[2] >= (list\_tissues[num1 - 1])[6] && loc2[2] <= (list\_tissues[num1 - 1])[3])) || ((loc2[1] >= (list\_tissues[num1 - 1])[5] && loc2[1] <= (list\_tissues[num1 - 1])[2]) && (loc1[2] >= (list\_tissues[num1 - 1])[6] && loc1[2] <= (list\_tissues[num1 - 1])[3]) || (loc1[1] >= (list\_tissues[num1 - 1])[5] && loc1[1] <= (list\_tissues[num1 - 1])[2]) && (loc1[2] >= (list\_tissues[num1 - 1])[6] && loc1[2] <= (list\_tissues[num1 - 1])[3]))) {

if (loc1[0] <= (list\_tissues[num1 - 1])[1]) {

if (loc2[0] >= (list\_tissues[num1 - 1])[4]) {

make = false;

}

}

}

if (((loc2[2] <= (list\_tissues[num1 - 1])[3] && loc2[2] >= (list\_tissues[num1 - 1])[6]) && (loc2[0] >= (list\_tissues[num1 - 1])[1] && loc2[0] <= (list\_tissues[num1 - 1])[4]) || (loc1[0] >= (list\_tissues[num1 - 1])[1] && loc1[0] <= (list\_tissues[num1 - 1])[4]) && (loc2[2] <= (list\_tissues[num1 - 1])[3] && loc2[2] >= (list\_tissues[num1 - 1])[6])) || ((loc1[2] <= (list\_tissues[num1 - 1])[3] && loc1[2] >= (list\_tissues[num1 - 1])[6]) && (loc1[0] >= (list\_tissues[num1 - 1])[1] && loc1[0] <= (list\_tissues[num1 - 1])[4]) || (loc1[2] <= (list\_tissues[num1 - 1])[3] && loc1[2] >= (list\_tissues[num1 - 1])[6]) && (loc2[0] >= (list\_tissues[num1 - 1])[1] && loc2[0] <= (list\_tissues[num1 - 1])[4]))) {

if (loc1[1] >= (list\_tissues[num1 - 1])[2]) {

if (loc2[1] <= (list\_tissues[num1 - 1])[5]) {

make = false;

}

}

}

if (((loc2[0] <= (list\_tissues[num1 - 1])[4] && loc2[0] >= (list\_tissues[num1 - 1])[1]) && (loc1[1] >= (list\_tissues[num1 - 1])[5] && loc1[1] <= (list\_tissues[num1 - 1])[2]) || (loc1[0] <= (list\_tissues[num1 - 1])[4] && loc1[0] >= (list\_tissues[num1 - 1])[1]) && (loc1[1] >= (list\_tissues[num1 - 1])[5] && loc1[1] <= (list\_tissues[num1 - 1])[2])) || ((loc1[0] <= (list\_tissues[num1 - 1])[4] && loc1[0] >= (list\_tissues[num1 - 1])[1]) && (loc2[1] >= (list\_tissues[num1 - 1])[5] && loc2[1] <= (list\_tissues[num1 - 1])[2]) || (loc2[0] <= (list\_tissues[num1 - 1])[4] && loc2[0] >= (list\_tissues[num1 - 1])[1]) && (loc2[1] >= (list\_tissues[num1 - 1])[5] && loc2[1] <= (list\_tissues[num1 - 1])[2]))) {

if (loc1[2] >= (list\_tissues[num1 - 1])[3]) {

if (loc2[2] <= (list\_tissues[num1 - 1])[6]) {

make = false;

}

}

}

}

num1 = num1 + 1;

}

}

function screen\_cells() {

num1 = 1;

var repeat\_end1578 = cells\_list.length;

for (var count1578 = 0; count1578 < repeat\_end1578; count1578++) {

yesno = false;

if (yesno == false) {

if ((cells\_list[num1 - 1])[1] >= loc1[0]) {

if ((cells\_list[num1 - 1])[1] <= loc2[0]) {

if ((cells\_list[num1 - 1])[2] <= loc1[1]) {

if ((cells\_list[num1 - 1])[2] >= loc2[1]) {

if ((cells\_list[num1 - 1])[3] <= loc1[2]) {

if ((cells\_list[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[4] >= loc1[0]) {

if ((cells\_list[num1 - 1])[4] <= loc2[0]) {

if ((cells\_list[num1 - 1])[5] <= loc1[1]) {

if ((cells\_list[num1 - 1])[5] >= loc2[1]) {

if ((cells\_list[num1 - 1])[6] <= loc1[2]) {

if ((cells\_list[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[4] >= loc1[0]) {

if ((cells\_list[num1 - 1])[4] <= loc2[0]) {

if ((cells\_list[num1 - 1])[2] <= loc1[1]) {

if ((cells\_list[num1 - 1])[2] >= loc2[1]) {

if ((cells\_list[num1 - 1])[3] <= loc1[2]) {

if ((cells\_list[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[1] >= loc1[0]) {

if ((cells\_list[num1 - 1])[1] <= loc2[0]) {

if ((cells\_list[num1 - 1])[5] <= loc1[1]) {

if ((cells\_list[num1 - 1])[5] >= loc2[1]) {

if ((cells\_list[num1 - 1])[3] <= loc1[2]) {

if ((cells\_list[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[1] >= loc1[0]) {

if ((cells\_list[num1 - 1])[1] <= loc2[0]) {

if ((cells\_list[num1 - 1])[2] <= loc1[1]) {

if ((cells\_list[num1 - 1])[2] >= loc2[1]) {

if ((cells\_list[num1 - 1])[6] <= loc1[2]) {

if ((cells\_list[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[1] >= loc1[0]) {

if ((cells\_list[num1 - 1])[1] <= loc2[0]) {

if ((cells\_list[num1 - 1])[5] <= loc1[1]) {

if ((cells\_list[num1 - 1])[5] >= loc2[1]) {

if ((cells\_list[num1 - 1])[6] <= loc1[2]) {

if ((cells\_list[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[4] >= loc1[0]) {

if ((cells\_list[num1 - 1])[4] <= loc2[0]) {

if ((cells\_list[num1 - 1])[2] <= loc1[1]) {

if ((cells\_list[num1 - 1])[2] >= loc2[1]) {

if ((cells\_list[num1 - 1])[6] <= loc1[2]) {

if ((cells\_list[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((cells\_list[num1 - 1])[4] >= loc1[0]) {

if ((cells\_list[num1 - 1])[4] <= loc2[0]) {

if ((cells\_list[num1 - 1])[5] <= loc1[1]) {

if ((cells\_list[num1 - 1])[5] >= loc2[1]) {

if ((cells\_list[num1 - 1])[3] <= loc1[2]) {

if ((cells\_list[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if (loc1[0] >= (cells\_list[num1 - 1])[1]) {

if (loc1[0] <= (cells\_list[num1 - 1])[4]) {

if (loc1[1] <= (cells\_list[num1 - 1])[2]) {

if (loc1[1] >= (cells\_list[num1 - 1])[5]) {

if (loc1[2] <= (cells\_list[num1 - 1])[3]) {

if (loc1[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (cells\_list[num1 - 1])[1]) {

if (loc2[0] <= (cells\_list[num1 - 1])[4]) {

if (loc2[1] <= (cells\_list[num1 - 1])[2]) {

if (loc2[1] >= (cells\_list[num1 - 1])[5]) {

if (loc2[2] <= (cells\_list[num1 - 1])[3]) {

if (loc2[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (cells\_list[num1 - 1])[1]) {

if (loc2[0] <= (cells\_list[num1 - 1])[4]) {

if (loc1[1] <= (cells\_list[num1 - 1])[2]) {

if (loc1[1] >= (cells\_list[num1 - 1])[5]) {

if (loc1[2] <= (cells\_list[num1 - 1])[3]) {

if (loc1[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (cells\_list[num1 - 1])[1]) {

if (loc1[0] <= (cells\_list[num1 - 1])[4]) {

if (loc2[1] <= (cells\_list[num1 - 1])[2]) {

if (loc2[1] >= (cells\_list[num1 - 1])[5]) {

if (loc1[2] <= (cells\_list[num1 - 1])[3]) {

if (loc1[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (cells\_list[num1 - 1])[1]) {

if (loc1[0] <= (cells\_list[num1 - 1])[4]) {

if (loc1[1] <= (cells\_list[num1 - 1])[2]) {

if (loc1[1] >= (cells\_list[num1 - 1])[5]) {

if (loc2[2] <= (cells\_list[num1 - 1])[3]) {

if (loc2[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (cells\_list[num1 - 1])[1]) {

if (loc1[0] <= (cells\_list[num1 - 1])[4]) {

if (loc2[1] <= (cells\_list[num1 - 1])[2]) {

if (loc2[1] >= (cells\_list[num1 - 1])[5]) {

if (loc2[2] <= (cells\_list[num1 - 1])[3]) {

if (loc2[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (cells\_list[num1 - 1])[1]) {

if (loc2[0] <= (cells\_list[num1 - 1])[4]) {

if (loc1[1] <= (cells\_list[num1 - 1])[2]) {

if (loc1[1] >= (cells\_list[num1 - 1])[5]) {

if (loc2[2] <= (cells\_list[num1 - 1])[3]) {

if (loc2[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (cells\_list[num1 - 1])[1]) {

if (loc2[0] <= (cells\_list[num1 - 1])[4]) {

if (loc2[1] <= (cells\_list[num1 - 1])[2]) {

if (loc2[1] >= (cells\_list[num1 - 1])[5]) {

if (loc1[2] <= (cells\_list[num1 - 1])[3]) {

if (loc1[2] >= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if ((((cells\_list[num1 - 1])[2] >= loc2[1] && (cells\_list[num1 - 1])[2] <= loc1[1]) && ((cells\_list[num1 - 1])[3] >= loc2[2] && (cells\_list[num1 - 1])[3] <= loc1[2]) || ((cells\_list[num1 - 1])[5] >= loc2[1] && (cells\_list[num1 - 1])[5] <= loc1[1]) && ((cells\_list[num1 - 1])[3] >= loc2[2] && (cells\_list[num1 - 1])[3] <= loc1[2])) || (((cells\_list[num1 - 1])[3] >= loc2[1] && (cells\_list[num1 - 1])[3] <= loc1[1]) && ((cells\_list[num1 - 1])[6] >= loc2[2] && (cells\_list[num1 - 1])[6] <= loc1[2]) || ((cells\_list[num1 - 1])[5] >= loc2[1] && (cells\_list[num1 - 1])[5] <= loc1[1]) && ((cells\_list[num1 - 1])[6] >= loc2[2] && (cells\_list[num1 - 1])[6] <= loc1[2]))) {

if ((cells\_list[num1 - 1])[1] <= loc1[0]) {

if ((cells\_list[num1 - 1])[4] >= loc2[0]) {

make = false;

}

}

}

if ((((cells\_list[num1 - 1])[3] <= loc1[2] && (cells\_list[num1 - 1])[3] >= loc2[2]) && ((cells\_list[num1 - 1])[1] >= loc1[0] && (cells\_list[num1 - 1])[1] <= loc2[0]) || ((cells\_list[num1 - 1])[4] >= loc1[0] && (cells\_list[num1 - 1])[4] <= loc2[0]) && ((cells\_list[num1 - 1])[3] <= loc1[2] && (cells\_list[num1 - 1])[3] >= loc2[2])) || (((cells\_list[num1 - 1])[1] >= loc1[0] && (cells\_list[num1 - 1])[1] <= loc2[0]) && ((cells\_list[num1 - 1])[6] <= loc1[2] && (cells\_list[num1 - 1])[6] >= loc2[2]) || ((cells\_list[num1 - 1])[4] >= loc1[0] && (cells\_list[num1 - 1])[4] <= loc2[0]) && ((cells\_list[num1 - 1])[6] <= loc1[2] && (cells\_list[num1 - 1])[6] >= loc2[2]))) {

if ((cells\_list[num1 - 1])[2] >= loc1[1]) {

if ((cells\_list[num1 - 1])[5] <= loc2[1]) {

make = false;

}

}

}

if ((((cells\_list[num1 - 1])[2] <= loc1[1] && (cells\_list[num1 - 1])[2] >= loc2[1]) && ((cells\_list[num1 - 1])[1] >= loc1[0] && (cells\_list[num1 - 1])[1] <= loc2[0]) || ((cells\_list[num1 - 1])[2] <= loc1[1] && (cells\_list[num1 - 1])[2] >= loc2[1]) && ((cells\_list[num1 - 1])[4] >= loc1[0] && (cells\_list[num1 - 1])[4] <= loc2[0])) || (((cells\_list[num1 - 1])[5] <= loc1[1] && (cells\_list[num1 - 1])[5] >= loc2[1]) && ((cells\_list[num1 - 1])[4] >= loc1[0] && (cells\_list[num1 - 1])[4] <= loc2[0]) || ((cells\_list[num1 - 1])[5] <= loc1[1] && (cells\_list[num1 - 1])[5] >= loc2[1]) && ((cells\_list[num1 - 1])[1] >= loc1[0] && (cells\_list[num1 - 1])[1] <= loc2[0]))) {

if ((cells\_list[num1 - 1])[3] >= loc1[2]) {

if ((cells\_list[num1 - 1])[6] <= loc2[2]) {

make = false;

}

}

}

if (((loc1[1] >= (cells\_list[num1 - 1])[5] && loc1[1] <= (cells\_list[num1 - 1])[2]) && (loc2[2] >= (cells\_list[num1 - 1])[6] && loc2[2] <= (cells\_list[num1 - 1])[3]) || (loc2[1] >= (cells\_list[num1 - 1])[5] && loc2[1] <= (cells\_list[num1 - 1])[2]) && (loc2[2] >= (cells\_list[num1 - 1])[6] && loc2[2] <= (cells\_list[num1 - 1])[3])) || ((loc2[1] >= (cells\_list[num1 - 1])[5] && loc2[1] <= (cells\_list[num1 - 1])[2]) && (loc1[2] >= (cells\_list[num1 - 1])[6] && loc1[2] <= (cells\_list[num1 - 1])[3]) || (loc1[1] >= (cells\_list[num1 - 1])[5] && loc1[1] <= (cells\_list[num1 - 1])[2]) && (loc1[2] >= (cells\_list[num1 - 1])[6] && loc1[2] <= (cells\_list[num1 - 1])[3]))) {

if (loc1[0] <= (cells\_list[num1 - 1])[1]) {

if (loc2[0] >= (cells\_list[num1 - 1])[4]) {

make = false;

}

}

}

if (((loc2[2] <= (cells\_list[num1 - 1])[3] && loc2[2] >= (cells\_list[num1 - 1])[6]) && (loc2[0] >= (cells\_list[num1 - 1])[1] && loc2[0] <= (cells\_list[num1 - 1])[4]) || (loc1[0] >= (cells\_list[num1 - 1])[1] && loc1[0] <= (cells\_list[num1 - 1])[4]) && (loc2[2] <= (cells\_list[num1 - 1])[3] && loc2[2] >= (cells\_list[num1 - 1])[6])) || ((loc1[2] <= (cells\_list[num1 - 1])[3] && loc1[2] >= (cells\_list[num1 - 1])[6]) && (loc1[0] >= (cells\_list[num1 - 1])[1] && loc1[0] <= (cells\_list[num1 - 1])[4]) || (loc1[2] <= (cells\_list[num1 - 1])[3] && loc1[2] >= (cells\_list[num1 - 1])[6]) && (loc2[0] >= (cells\_list[num1 - 1])[1] && loc2[0] <= (cells\_list[num1 - 1])[4]))) {

if (loc1[1] >= (cells\_list[num1 - 1])[2]) {

if (loc2[1] <= (cells\_list[num1 - 1])[5]) {

make = false;

}

}

}

if (((loc2[0] <= (cells\_list[num1 - 1])[4] && loc2[0] >= (cells\_list[num1 - 1])[1]) && (loc1[1] >= (cells\_list[num1 - 1])[5] && loc1[1] <= (cells\_list[num1 - 1])[2]) || (loc1[0] <= (cells\_list[num1 - 1])[4] && loc1[0] >= (cells\_list[num1 - 1])[1]) && (loc1[1] >= (cells\_list[num1 - 1])[5] && loc1[1] <= (cells\_list[num1 - 1])[2])) || ((loc1[0] <= (cells\_list[num1 - 1])[4] && loc1[0] >= (cells\_list[num1 - 1])[1]) && (loc2[1] >= (cells\_list[num1 - 1])[5] && loc2[1] <= (cells\_list[num1 - 1])[2]) || (loc2[0] <= (cells\_list[num1 - 1])[4] && loc2[0] >= (cells\_list[num1 - 1])[1]) && (loc2[1] >= (cells\_list[num1 - 1])[5] && loc2[1] <= (cells\_list[num1 - 1])[2]))) {

if (loc1[2] >= (cells\_list[num1 - 1])[3]) {

if (loc2[2] <= (cells\_list[num1 - 1])[6]) {

make = false;

}

}

}

}

num1 = num1 + 1;

}

}

function screen\_molecules() {

num1 = 1;

var repeat\_end1579 = list\_molecules.length;

for (var count1579 = 0; count1579 < repeat\_end1579; count1579++) {

yesno = false;

if (yesno == false) {

if ((list\_molecules[num1 - 1])[1] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[1] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[2] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[2] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[3] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[4] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[4] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[5] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[5] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[6] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[4] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[4] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[2] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[2] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[3] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[1] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[1] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[5] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[5] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[3] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[1] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[1] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[2] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[2] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[6] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[1] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[1] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[5] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[5] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[6] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[4] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[4] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[2] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[2] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[6] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[6] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if ((list\_molecules[num1 - 1])[4] >= loc1[0]) {

if ((list\_molecules[num1 - 1])[4] <= loc2[0]) {

if ((list\_molecules[num1 - 1])[5] <= loc1[1]) {

if ((list\_molecules[num1 - 1])[5] >= loc2[1]) {

if ((list\_molecules[num1 - 1])[3] <= loc1[2]) {

if ((list\_molecules[num1 - 1])[3] >= loc2[2]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if (loc1[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc1[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc1[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc1[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc1[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc1[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc2[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc2[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc2[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc2[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc2[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc2[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc1[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc1[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc1[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc1[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc1[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc2[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc2[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc1[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc1[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc1[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc1[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc1[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc2[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc2[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc1[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc2[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc2[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc2[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc2[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc2[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc1[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc1[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc2[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc2[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= (list\_molecules[num1 - 1])[1]) {

if (loc2[0] <= (list\_molecules[num1 - 1])[4]) {

if (loc2[1] <= (list\_molecules[num1 - 1])[2]) {

if (loc2[1] >= (list\_molecules[num1 - 1])[5]) {

if (loc1[2] <= (list\_molecules[num1 - 1])[3]) {

if (loc1[2] >= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if ((((list\_molecules[num1 - 1])[2] >= loc2[1] && (list\_molecules[num1 - 1])[2] <= loc1[1]) && ((list\_molecules[num1 - 1])[3] >= loc2[2] && (list\_molecules[num1 - 1])[3] <= loc1[2]) || ((list\_molecules[num1 - 1])[5] >= loc2[1] && (list\_molecules[num1 - 1])[5] <= loc1[1]) && ((list\_molecules[num1 - 1])[3] >= loc2[2] && (list\_molecules[num1 - 1])[3] <= loc1[2])) || (((list\_molecules[num1 - 1])[3] >= loc2[1] && (list\_molecules[num1 - 1])[3] <= loc1[1]) && ((list\_molecules[num1 - 1])[6] >= loc2[2] && (list\_molecules[num1 - 1])[6] <= loc1[2]) || ((list\_molecules[num1 - 1])[5] >= loc2[1] && (list\_molecules[num1 - 1])[5] <= loc1[1]) && ((list\_molecules[num1 - 1])[6] >= loc2[2] && (list\_molecules[num1 - 1])[6] <= loc1[2]))) {

if ((list\_molecules[num1 - 1])[1] <= loc1[0]) {

if ((list\_molecules[num1 - 1])[4] >= loc2[0]) {

make = false;

}

}

}

if ((((list\_molecules[num1 - 1])[3] <= loc1[2] && (list\_molecules[num1 - 1])[3] >= loc2[2]) && ((list\_molecules[num1 - 1])[1] >= loc1[0] && (list\_molecules[num1 - 1])[1] <= loc2[0]) || ((list\_molecules[num1 - 1])[4] >= loc1[0] && (list\_molecules[num1 - 1])[4] <= loc2[0]) && ((list\_molecules[num1 - 1])[3] <= loc1[2] && (list\_molecules[num1 - 1])[3] >= loc2[2])) || (((list\_molecules[num1 - 1])[1] >= loc1[0] && (list\_molecules[num1 - 1])[1] <= loc2[0]) && ((list\_molecules[num1 - 1])[6] <= loc1[2] && (list\_molecules[num1 - 1])[6] >= loc2[2]) || ((list\_molecules[num1 - 1])[4] >= loc1[0] && (list\_molecules[num1 - 1])[4] <= loc2[0]) && ((list\_molecules[num1 - 1])[6] <= loc1[2] && (list\_molecules[num1 - 1])[6] >= loc2[2]))) {

if ((list\_molecules[num1 - 1])[2] >= loc1[1]) {

if ((list\_molecules[num1 - 1])[5] <= loc2[1]) {

make = false;

}

}

}

if ((((list\_molecules[num1 - 1])[2] <= loc1[1] && (list\_molecules[num1 - 1])[2] >= loc2[1]) && ((list\_molecules[num1 - 1])[1] >= loc1[0] && (list\_molecules[num1 - 1])[1] <= loc2[0]) || ((list\_molecules[num1 - 1])[2] <= loc1[1] && (list\_molecules[num1 - 1])[2] >= loc2[1]) && ((list\_molecules[num1 - 1])[4] >= loc1[0] && (list\_molecules[num1 - 1])[4] <= loc2[0])) || (((list\_molecules[num1 - 1])[5] <= loc1[1] && (list\_molecules[num1 - 1])[5] >= loc2[1]) && ((list\_molecules[num1 - 1])[4] >= loc1[0] && (list\_molecules[num1 - 1])[4] <= loc2[0]) || ((list\_molecules[num1 - 1])[5] <= loc1[1] && (list\_molecules[num1 - 1])[5] >= loc2[1]) && ((list\_molecules[num1 - 1])[1] >= loc1[0] && (list\_molecules[num1 - 1])[1] <= loc2[0]))) {

if ((list\_molecules[num1 - 1])[3] >= loc1[2]) {

if ((list\_molecules[num1 - 1])[6] <= loc2[2]) {

make = false;

}

}

}

if (((loc1[1] >= (list\_molecules[num1 - 1])[5] && loc1[1] <= (list\_molecules[num1 - 1])[2]) && (loc2[2] >= (list\_molecules[num1 - 1])[6] && loc2[2] <= (list\_molecules[num1 - 1])[3]) || (loc2[1] >= (list\_molecules[num1 - 1])[5] && loc2[1] <= (list\_molecules[num1 - 1])[2]) && (loc2[2] >= (list\_molecules[num1 - 1])[6] && loc2[2] <= (list\_molecules[num1 - 1])[3])) || ((loc2[1] >= (list\_molecules[num1 - 1])[5] && loc2[1] <= (list\_molecules[num1 - 1])[2]) && (loc1[2] >= (list\_molecules[num1 - 1])[6] && loc1[2] <= (list\_molecules[num1 - 1])[3]) || (loc1[1] >= (list\_molecules[num1 - 1])[5] && loc1[1] <= (list\_molecules[num1 - 1])[2]) && (loc1[2] >= (list\_molecules[num1 - 1])[6] && loc1[2] <= (list\_molecules[num1 - 1])[3]))) {

if (loc1[0] <= (list\_molecules[num1 - 1])[1]) {

if (loc2[0] >= (list\_molecules[num1 - 1])[4]) {

make = false;

}

}

}

if (((loc2[2] <= (list\_molecules[num1 - 1])[3] && loc2[2] >= (list\_molecules[num1 - 1])[6]) && (loc2[0] >= (list\_molecules[num1 - 1])[1] && loc2[0] <= (list\_molecules[num1 - 1])[4]) || (loc1[0] >= (list\_molecules[num1 - 1])[1] && loc1[0] <= (list\_molecules[num1 - 1])[4]) && (loc2[2] <= (list\_molecules[num1 - 1])[3] && loc2[2] >= (list\_molecules[num1 - 1])[6])) || ((loc1[2] <= (list\_molecules[num1 - 1])[3] && loc1[2] >= (list\_molecules[num1 - 1])[6]) && (loc1[0] >= (list\_molecules[num1 - 1])[1] && loc1[0] <= (list\_molecules[num1 - 1])[4]) || (loc1[2] <= (list\_molecules[num1 - 1])[3] && loc1[2] >= (list\_molecules[num1 - 1])[6]) && (loc2[0] >= (list\_molecules[num1 - 1])[1] && loc2[0] <= (list\_molecules[num1 - 1])[4]))) {

if (loc1[1] >= (list\_molecules[num1 - 1])[2]) {

if (loc2[1] <= (list\_molecules[num1 - 1])[5]) {

make = false;

}

}

}

if (((loc2[0] <= (list\_molecules[num1 - 1])[4] && loc2[0] >= (list\_molecules[num1 - 1])[1]) && (loc1[1] >= (list\_molecules[num1 - 1])[5] && loc1[1] <= (list\_molecules[num1 - 1])[2]) || (loc1[0] <= (list\_molecules[num1 - 1])[4] && loc1[0] >= (list\_molecules[num1 - 1])[1]) && (loc1[1] >= (list\_molecules[num1 - 1])[5] && loc1[1] <= (list\_molecules[num1 - 1])[2])) || ((loc1[0] <= (list\_molecules[num1 - 1])[4] && loc1[0] >= (list\_molecules[num1 - 1])[1]) && (loc2[1] >= (list\_molecules[num1 - 1])[5] && loc2[1] <= (list\_molecules[num1 - 1])[2]) || (loc2[0] <= (list\_molecules[num1 - 1])[4] && loc2[0] >= (list\_molecules[num1 - 1])[1]) && (loc2[1] >= (list\_molecules[num1 - 1])[5] && loc2[1] <= (list\_molecules[num1 - 1])[2]))) {

if (loc1[2] >= (list\_molecules[num1 - 1])[3]) {

if (loc2[2] <= (list\_molecules[num1 - 1])[6]) {

make = false;

}

}

}

}

num1 = num1 + 1;

}

}

function box\_2() {

yesno = false;

if (yesno == false) {

if (box2[0] >= loc1[0]) {

if (box2[0] <= loc2[0]) {

if (box2[1] <= loc1[1]) {

if (box2[1] >= loc2[1]) {

if (box2[2] <= loc1[2]) {

if (box2[2] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[3] >= loc1[0]) {

if (box2[3] <= loc2[0]) {

if (box2[4] <= loc1[1]) {

if (box2[4] >= loc2[1]) {

if (box2[5] <= loc1[2]) {

if (box2[5] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[3] >= loc1[0]) {

if (box2[3] <= loc2[0]) {

if (box2[1] <= loc1[1]) {

if (box2[1] >= loc2[1]) {

if (box2[2] <= loc1[2]) {

if (box2[2] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[0] >= loc1[0]) {

if (box2[0] <= loc2[0]) {

if (box2[4] <= loc1[1]) {

if (box2[4] >= loc2[1]) {

if (box2[2] <= loc1[2]) {

if (box2[2] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[0] >= loc1[0]) {

if (box2[0] <= loc2[0]) {

if (box2[1] <= loc1[1]) {

if (box2[1] >= loc2[1]) {

if (box2[5] <= loc1[2]) {

if (box2[5] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[0] >= loc1[0]) {

if (box2[0] <= loc2[0]) {

if (box2[4] <= loc1[1]) {

if (box2[4] >= loc2[1]) {

if (box2[5] <= loc1[2]) {

if (box2[5] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[3] >= loc1[0]) {

if (box2[3] <= loc2[0]) {

if (box2[1] <= loc1[1]) {

if (box2[1] >= loc2[1]) {

if (box2[5] <= loc1[2]) {

if (box2[5] >= loc2[2]) {

make = false;

}

}

}

}

}

}

if (box2[3] >= loc1[0]) {

if (box2[3] <= loc2[0]) {

if (box2[4] <= loc1[1]) {

if (box2[4] >= loc2[1]) {

if (box2[1] <= loc1[2]) {

if (box2[1] >= loc2[2]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if (loc1[0] >= box2[0]) {

if (loc1[0] <= box2[3]) {

if (loc1[1] <= box2[1]) {

if (loc1[1] >= box2[4]) {

if (loc1[2] <= box2[2]) {

if (loc1[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= box2[0]) {

if (loc2[0] <= box2[3]) {

if (loc2[1] <= box2[1]) {

if (loc2[1] >= box2[4]) {

if (loc2[2] <= box2[2]) {

if (loc2[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= box2[0]) {

if (loc2[0] <= box2[3]) {

if (loc1[1] <= box2[1]) {

if (loc1[1] >= box2[4]) {

if (loc1[2] <= box2[2]) {

if (loc1[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= box2[0]) {

if (loc1[0] <= box2[3]) {

if (loc2[1] <= box2[1]) {

if (loc2[1] >= box2[4]) {

if (loc1[2] <= box2[2]) {

if (loc1[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= box2[0]) {

if (loc1[0] <= box2[3]) {

if (loc1[1] <= box2[1]) {

if (loc1[1] >= box2[4]) {

if (loc2[2] <= box2[2]) {

if (loc2[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc1[0] >= box2[0]) {

if (loc1[0] <= box2[3]) {

if (loc2[1] <= box2[1]) {

if (loc2[1] >= box2[4]) {

if (loc2[2] <= box2[2]) {

if (loc2[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= box2[0]) {

if (loc2[0] <= box2[3]) {

if (loc1[1] <= box2[1]) {

if (loc1[1] >= box2[4]) {

if (loc2[2] <= box2[2]) {

if (loc2[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

if (loc2[0] >= box2[0]) {

if (loc2[0] <= box2[3]) {

if (loc2[1] <= box2[1]) {

if (loc2[1] >= box2[4]) {

if (loc1[2] <= box2[2]) {

if (loc1[2] >= box2[5]) {

make = false;

}

}

}

}

}

}

}

if (yesno == false) {

if (((box2[1] >= loc2[1] && box2[1] <= loc1[1]) && (box2[2] >= loc2[2] && box2[2] <= loc1[2]) || (box2[4] >= loc2[1] && box2[4] <= loc1[1]) && (box2[2] >= loc2[2] && box2[2] <= loc1[2])) || ((box2[2] >= loc2[1] && box2[2] <= loc1[1]) && (box2[5] >= loc2[2] && box2[5] <= loc1[2]) || (box2[4] >= loc2[1] && box2[4] <= loc1[1]) && (box2[5] >= loc2[2] && box2[5] <= loc1[2]))) {

if (box2[0] <= loc1[0]) {

if (box2[3] >= loc2[0]) {

make = false;

}

}

}

if (((box2[2] <= loc1[2] && box2[2] >= loc2[2]) && (box2[0] >= loc1[0] && box2[0] <= loc2[0]) || (box2[3] >= loc1[0] && box2[3] <= loc2[0]) && (box2[2] <= loc1[2] && box2[2] >= loc2[2])) || ((box2[0] >= loc1[0] && box2[0] <= loc2[0]) && (box2[5] <= loc1[2] && box2[5] >= loc2[2]) || (box2[3] >= loc1[0] && box2[3] <= loc2[0]) && (box2[5] <= loc1[2] && box2[5] >= loc2[2]))) {

if (box2[1] >= loc1[1]) {

if (box2[4] <= loc2[1]) {

make = false;

}

}

}

if (((box2[1] <= loc1[1] && box2[1] >= loc2[1]) && (box2[0] >= loc1[0] && box2[0] <= loc2[0]) || (box2[1] <= loc1[1] && box2[1] >= loc2[1]) && (box2[3] >= loc1[0] && box2[3] <= loc2[0])) || ((box2[4] <= loc1[1] && box2[4] >= loc2[1]) && (box2[3] >= loc1[0] && box2[3] <= loc2[0]) || (box2[4] <= loc1[1] && box2[4] >= loc2[1]) && (box2[0] >= loc1[0] && box2[0] <= loc2[0]))) {

if (box2[2] >= loc1[2]) {

if (box2[5] <= loc2[2]) {

make = false;

}

}

}

if (((loc1[1] >= box2[4] && loc1[1] <= box2[1]) && (loc2[2] >= box2[5] && loc2[2] <= box2[2]) || (loc2[1] >= box2[4] && loc2[1] <= box2[1]) && (loc2[2] >= box2[5] && loc2[2] <= box2[2])) || ((loc2[1] >= box2[4] && loc2[1] <= box2[1]) && (loc1[2] >= box2[5] && loc1[2] <= box2[2]) || (loc1[1] >= box2[4] && loc1[1] <= box2[1]) && (loc1[2] >= box2[5] && loc1[2] <= box2[2]))) {

if (loc1[0] <= box2[0]) {

if (loc2[0] >= box2[3]) {

make = false;

}

}

}

if (((loc2[2] <= box2[2] && loc2[2] >= box2[5]) && (loc2[0] >= box2[0] && loc2[0] <= box2[3]) || (loc1[0] >= box2[0] && loc1[0] <= box2[3]) && (loc2[2] <= box2[2] && loc2[2] >= box2[5])) || ((loc1[2] <= box2[2] && loc1[2] >= box2[5]) && (loc1[0] >= box2[0] && loc1[0] <= box2[3]) || (loc1[2] <= box2[2] && loc1[2] >= box2[5]) && (loc2[0] >= box2[0] && loc2[0] <= box2[3]))) {

if (loc1[1] >= box2[1]) {

if (loc2[1] <= box2[4]) {

make = false;

}

}

}

if (((loc2[0] <= box2[3] && loc2[0] >= box2[0]) && (loc1[1] >= box2[4] && loc1[1] <= box2[1]) || (loc1[0] <= box2[3] && loc1[0] >= box2[0]) && (loc1[1] >= box2[4] && loc1[1] <= box2[1])) || ((loc1[0] <= box2[3] && loc1[0] >= box2[0]) && (loc2[1] >= box2[4] && loc2[1] <= box2[1]) || (loc2[0] <= box2[3] && loc2[0] >= box2[0]) && (loc2[1] >= box2[4] && loc2[1] <= box2[1]))) {

if (loc1[2] >= box2[2]) {

if (loc2[2] <= box2[5]) {

make = false;

}

}

}

}

}

function spiracle() {

made = false;

spirnum.push(list\_tissues.length);

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else {

spiracle\_2();

}

}

}

}

spirnum.pop();

}

function spiracle\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = true;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.949) {

if (Math.abs(Math.pow(Math.abs(loc2[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc2[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(Math.pow(Math.abs(loc1[1] - (centerpoint\_y - sizemin \* 0.5)), 2) + Math.pow(Math.abs(loc1[2] - (centerpoint\_x - sizemid \* 0.5)), 2)) < sizemin \* 0.49) {

if (Math.abs(loc1[0] - loc2[0]) >= 2.475170941e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 2.475170941e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 2.475170941e+29) {

list\_tissues.push(['spiracle', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

spiracle();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

}

function duct() {

made = false;

spirnum.push(list\_tissues.length);

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else {

duct\_2();

}

}

}

}

spirnum.pop();

}

function duct\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['duct', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

duct();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

}

function lymph() {

made = false;

spirnum.push(list\_tissues.length);

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

} else {

lymph\_2();

}

}

}

}

spirnum.pop();

}

function lymph\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999;

spirlength = 10;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+30) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+30) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+30) {

list\_tissues.push(['lymph', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

lymph();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999;

spirlength = spirlength \* 0.999;

}

}

}

function vein() {

made = false;

spirnum.push(list\_tissues.length);

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0] + 1000000000000, loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capilary', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1] + 1000000000000, loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1] + 1000000000000, loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2] + 1000000000000]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

vein\_2();

} else {

vein\_3();

}

}

}

}

spirnum.pop();

}

function vein\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function vein\_3() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['venule', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['vein', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

vein();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function artery() {

made = false;

spirnum.push(list\_tissues.length);

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) >= 1.23758547e+30) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if ((Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30) || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30) || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0] - 1000000000000, loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

artery\_2();

} else {

artery\_3();

}

}

}

}

spirnum.pop();

}

function artery\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function artery\_3() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

exempt = ['alveolar capillary', 'capillary'];

} else {

exempt = [null];

}

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

list\_tissues.push(['capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else if (Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+30 || (Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+30 || Math.abs(loc1[0] - loc2[0]) <= 3.093963676e+30)) {

list\_tissues.push(['arteriole', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

} else {

list\_tissues.push(['artery', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

artery();

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function lung() {

made = false;

spirnum.push(list\_tissues.length);

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

made = true;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] - 1000000000000, loc4[1] - 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc3[1] + 1000000000000, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] + Math.abs(loc3[0] - loc4[0]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - Math.abs(loc3[0] - loc4[0]) \* spirlength, loc4[2] - 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc2[2] + Math.abs(loc3[1] - loc4[1]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 0, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - 0, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc3[0] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc4[1] - 1000000000000, loc3[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirwidth, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if ((Math.abs(loc1[2] - loc2[2]) <= 3.093963676e+31 || Math.abs(loc1[1] - loc2[1]) <= 3.093963676e+31) || Math.abs(loc1[0] - loc2[0]) >= 3.093963676e+31) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

lung\_2();

} else {

lung\_3();

}

spirnum.pop();

}

function lung\_2() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc2 = [loc4[0] - 1000000000000, loc3[1] + 1000000000000, loc4[2] + 1000000000000];

loc1 = [loc2[0] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc2[1] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc2[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

if (Math.abs(loc1[0] - loc2[0]) >= 6.187927353e+29) {

if (Math.abs(loc1[1] - loc2[1]) >= 6.187927353e+29) {

if (Math.abs(loc1[2] - loc2[2]) >= 6.187927353e+29) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

}

}

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc4[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[1] - loc4[1]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] - Math.abs(loc3[1] - loc4[1]) \* spirlength, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + Math.abs(loc3[1] - loc4[1]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

loc1 = [loc3[0] + 1000000000000, loc4[1] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[1] - loc4[1]) \* spirlength];

exempt = [null];

make = false;

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function lung\_3() {

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[2] - loc4[2]) \* spirlength];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] + Math.abs(loc3[2] - loc4[2]) \* spirlength];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[2] + 1000000000000];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc4[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[1] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] + 1000000000000, loc3[1] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc3[1] + 1000000000000, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc4[0] + 1000000000000, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc1[0] + Math.abs(loc3[2] - loc4[2]) \* spirlength, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc3[2] - 1000000000000];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

if (unused\_variable == 5) {

bool = true;

spirwidth = 0.999999;

spirlength = math\_random\_int(5e+33, 1e+34) / 1e+33;

while (bool == true && spirlength >= 0.25) {

while (spirwidth >= 0.001 && bool == true) {

make = false;

loc1 = [loc3[0] - Math.abs(loc3[2] - loc4[2]) \* spirlength, loc3[1] - 1000000000000, loc4[2] + Math.abs(loc3[0] - loc4[0]) \* spirwidth];

loc2 = [loc3[0] - 1000000000000, loc1[1] - Math.abs(loc3[0] - loc4[0]) \* spirwidth, loc1[2] - Math.abs(loc3[0] - loc4[0]) \* spirwidth];

exempt = [null];

box3();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

}

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['bronch', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

if (Math.abs(loc1[2] - loc2[2]) <= 1.23758547e+30 || (Math.abs(loc1[1] - loc2[1]) <= 1.23758547e+30 || Math.abs(loc1[0] - loc2[0]) <= 1.23758547e+30)) {

alveolus();

} else {

lung();

}

loc1 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[1], (list\_tissues[spirnum.slice(-1)[0] - 1])[2], (list\_tissues[spirnum.slice(-1)[0] - 1])[3]];

loc2 = [(list\_tissues[spirnum.slice(-1)[0] - 1])[4], (list\_tissues[spirnum.slice(-1)[0] - 1])[5], (list\_tissues[spirnum.slice(-1)[0] - 1])[6]];

loc3 = loc1;

loc4 = loc2;

bool = false;

}

spirwidth = spirwidth \* 0.999;

}

spirwidth = 0.999999;

spirlength = spirlength \* (math\_random\_int(8e+32, 9e+32) / 1e+33);

}

}

}

function alveolus() {

made = false;

spirnum.push(list\_tissues.length);

loc3 = loc1;

loc4 = loc2;

if (Math.abs(loc1[0] - loc2[0]) > Math.abs(loc1[1] - loc2[1])) {

if (unused\_variable == 5) {

cursor1x = loc3[2];

cursor1y = loc3[1];

cursor1z = loc4[0] + 1000000000000;

alveolus\_2();

}

if (unused\_variable == 5) {

cursor1x = loc3[2];

cursor1y = loc3[1];

cursor1z = loc3[0] - 3.093963676e+31;

alveolus\_2();

}

} else if (Math.abs(loc1[1] - loc2[1]) > Math.abs(loc1[2] - loc2[2])) {

if (unused\_variable == 5) {

cursor1x = loc3[2];

cursor1y = loc4[1] - 1000000000000;

cursor1z = loc3[0];

alveolus\_2();

}

if (unused\_variable == 5) {

cursor1x = loc3[2];

cursor1y = loc3[1] + 3.093963676e+31;

cursor1z = loc3[0];

alveolus\_2();

}

} else {

if (unused\_variable == 5) {

cursor1x = loc4[2] - 1000000000000;

cursor1y = loc3[1];

cursor1z = loc3[0];

alveolus\_2();

}

if (unused\_variable == 5) {

cursor1x = loc3[2] + 1.23758547e+31;

cursor1y = loc3[1];

cursor1z = loc3[0];

alveolus\_2();

}

}

spirnum.pop();

}

function alveolus\_2() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

for (var count1580 = 0; count1580 < 50; count1580++) {

for (var count1581 = 0; count1581 < 50; count1581++) {

for (var count1582 = 0; count1582 < 20; count1582++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolus', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29;

}

cursor1x = anchorx;

cursor1y = cursor1y - 6.187927353e+29;

}

cursor1y = anchory;

cursor1z = cursor1z + 6.187927353e+29;

}

cursor1x = anchorx + 6.187927353e+30;

cursor1y = anchory;

cursor1z = anchorz;

for (var count1583 = 0; count1583 < 50; count1583++) {

for (var count1584 = 0; count1584 < 26; count1584++) {

for (var count1585 = 0; count1585 < 41; count1585++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolus', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29;

}

cursor1x = anchorx + 6.187927353e+30;

cursor1y = cursor1y - 6.187927353e+29 \* 2;

}

cursor1y = anchory;

cursor1z = cursor1z + 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = anchory + 6.187927353e+29;

cursor1z = anchorz - 6.187927353e+29;

for (var count1586 = 0; count1586 < 52; count1586++) {

for (var count1587 = 0; count1587 < 27; count1587++) {

for (var count1588 = 0; count1588 < 22; count1588++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y - 1.856378206e+28, cursor1x - 1.856378206e+28];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.002289533e+29, cursor1x - 6.002289533e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolar capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29 \* 2;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = cursor1y - 6.187927353e+29 \* 2;

}

cursor1y = anchory + 6.187927353e+29;

cursor1z = cursor1z + 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = anchory + 6.187927353e+29;

cursor1z = anchorz - 1.23758547e+30;

for (var count1589 = 0; count1589 < 54; count1589++) {

for (var count1590 = 0; count1590 < 22; count1590++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolar capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29 \* 2;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = cursor1y - 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = anchory + 6.187927353e+29;

cursor1z = anchorz - 1.23758547e+30;

for (var count1591 = 0; count1591 < 27; count1591++) {

for (var count1592 = 0; count1592 < 44; count1592++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolar capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = cursor1y - 6.187927353e+29 \* 2;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = anchory + 6.187927353e+29;

cursor1z = anchorz + 3.15584295e+31;

for (var count1593 = 0; count1593 < 54; count1593++) {

for (var count1594 = 0; count1594 < 22; count1594++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolar capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29 \* 2;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = cursor1y - 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = anchory + 6.187927353e+29;

cursor1z = anchorz + 3.15584295e+31;

for (var count1595 = 0; count1595 < 27; count1595++) {

for (var count1596 = 0; count1596 < 44; count1596++) {

if (unused\_variable == 5) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 6.187927352e+29, cursor1y - 6.187927352e+29, cursor1x - 6.187927352e+29];

make = false;

box3();

box\_2();

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

list\_tissues.push(['alveolar capillary', loc1[0], loc1[1], loc1[2], loc2[0], loc2[1], loc2[2]]);

}

}

cursor1x = cursor1x - 6.187927353e+29;

}

cursor1x = anchorx + 6.806720089e+30;

cursor1y = cursor1y - 6.187927353e+29 \* 2;

}

}

function invertebrate\_endoskeleton() {

if (cursor1y >= centerpoint\_y - sizemin \* 0.5) {

if (cursor1y <= centerpoint\_y - sizemin \* 0.4) {

if (cursor1x <= centerpoint\_x - sizemid \* 0.05) {

if (cursor1x >= centerpoint\_x - sizemid \* 0.15) {

if (cursor1z <= centerpoint\_z + sizemax \* 0.07) {

collagen();

} else if (cursor1z >= centerpoint\_z + sizemax \* 0.8) {

collagen();

}

}

} else if (cursor1x <= centerpoint\_x - sizemid \* 0.85) {

if (cursor1x >= centerpoint\_x - sizemid \* 0.95) {

if (cursor1z <= centerpoint\_z + sizemax \* 0.07) {

collagen();

} else if (cursor1z >= centerpoint\_z + sizemax \* 0.8) {

collagen();

}

}

}

}

} else if (cursor1y >= centerpoint\_y - sizemin \* 0.3) {

if (cursor1x <= centerpoint\_x - sizemid \* 0.8) {

if (cursor1z >= centerpoint\_z + sizemax \* 0.78) {

collagen();

} else if (cursor1z <= centerpoint\_z + sizemax \* 0.5) {

if (cursor1z <= centerpoint\_z + sizemax \* 0.389) {

collagen();

}

} else if (cursor1z <= centerpoint\_z + sizemax \* 0.115) {

collagen();

}

} else if (cursor1x >= centerpoint\_x - sizemid \* 0.2) {

if (cursor1z >= centerpoint\_z + sizemax \* 0.78) {

collagen();

} else if (cursor1z <= centerpoint\_z + sizemax \* 0.5) {

if (cursor1z <= centerpoint\_z + sizemax \* 0.389) {

collagen();

}

} else if (cursor1z <= centerpoint\_z + sizemax \* 0.115) {

collagen();

}

}

}

}

function teeth\_1() {

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.31) {

if (cursor1y > centerpoint\_y - sizemin \* 0.315) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.314) {

if (cursor1y > centerpoint\_y - sizemin \* 0.38) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.379) {

if (cursor1y > centerpoint\_y - sizemin \* 0.41) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.4) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.43) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.46) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.003) {

if (cursor1y < centerpoint\_y - sizemin \* 0.445) {

if (cursor1y > centerpoint\_y - sizemin \* 0.448) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.002) {

if (cursor1y < centerpoint\_y - sizemin \* 0.44) {

if (cursor1y > centerpoint\_y - sizemin \* 0.446) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.983) - cursor1z, 2) + Math.pow((centerpoint\_x - sizemid \* 0.58) - cursor1x, 2))) < sizemax \* 0.009) {

if (cursor1y < centerpoint\_y - sizemin \* 0.415) {

if (cursor1y > centerpoint\_y - sizemin \* 0.441) {

make = true;

}

}

}

}

}

function teeth\_2() {

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.48) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.938) {

if (cursor1z > cursor1z + sizemin \* 0.933) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.48) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.934) {

if (cursor1z > cursor1z + sizemin \* 0.92) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.938) {

if (cursor1z > cursor1z + sizemin \* 0.933) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.934) {

if (cursor1z > cursor1z + sizemin \* 0.92) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.938) {

if (cursor1z > cursor1z + sizemin \* 0.933) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.934) {

if (cursor1z > cursor1z + sizemin \* 0.92) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.51) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.938) {

if (cursor1z > cursor1z + sizemin \* 0.933) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.51) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.934) {

if (cursor1z > cursor1z + sizemin \* 0.92) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.938) {

if (cursor1z > cursor1z + sizemin \* 0.933) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.934) {

if (cursor1z > cursor1z + sizemin \* 0.92) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.48) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.907) {

if (cursor1z > cursor1z + sizemin \* 0.904) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.48) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.917) {

if (cursor1z > cursor1z + sizemin \* 0.906) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.907) {

if (cursor1z > cursor1z + sizemin \* 0.904) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.49) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.917) {

if (cursor1z > cursor1z + sizemin \* 0.906) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.907) {

if (cursor1z > cursor1z + sizemin \* 0.904) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.917) {

if (cursor1z > cursor1z + sizemin \* 0.906) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.51) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.907) {

if (cursor1z > cursor1z + sizemin \* 0.904) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.51) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.917) {

if (cursor1z > cursor1z + sizemin \* 0.906) {

make = true;

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemin \* 0.01) {

if (cursor1z < centerpoint\_z + sizemin \* 0.907) {

if (cursor1z > cursor1z + sizemin \* 0.904) {

make = true;

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - cursor1y, 2) + Math.pow((centerpoint\_x - sizemid \* 0.52) - cursor1x, 2))) < sizemin \* 0.009) {

if (cursor1z < centerpoint\_z + sizemin \* 0.917) {

if (cursor1z > cursor1z + sizemin \* 0.906) {

make = true;

}

}

}

}

}

function vertebrate\_body() {

if (loc1[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.8) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.75) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.22) - loc1[2], 2))) < sizemin \* 0.02) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.17) {

vertebrate\_body\_2();

}

} else {

if (loc1[2] < centerpoint\_x - sizemid \* 0.4775018) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.5205) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.4995) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.5205) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.2495) {

vertebrate\_body\_2();

}

}

}

}

}

}

}

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

vertebrate\_body\_2();

} else if (loc1[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.02) {

vertebrate\_body\_2();

} else if (loc1[2] > centerpoint\_x - sizemid \* 0.21) {

vertebrate\_body\_2();

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.79) {

vertebrate\_body\_2();

} else if (loc1[0] > centerpoint\_z + sizemax \* 0.74) {

vertebrate\_body\_2();

} else {

if (loc1[2] < centerpoint\_x - sizemid \* 0.65) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.35) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.65) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.1) {

vertebrate\_body\_2();

} else {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.37) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.001) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.4) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.0015) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.003) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.45) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.006) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.46) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.01) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.48) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.02) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.5) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.04) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.54) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.0005) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.56) - loc1[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.00025) {

vertebrate\_body\_2();

}

}

}

}

}

}

}

}

}

} else if (loc1[2] > centerpoint\_x - sizemid \* 0.99) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.3) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.37) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.81) {

vertebrate\_body\_2();

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.8) {

vertebrate\_body\_2();

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.435) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.935) {

vertebrate\_body\_2();

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.815) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.875) {

vertebrate\_body\_2();

}

}

}

} else {

if (loc1[0] > centerpoint\_z + sizemax \* 0.615) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.685) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.81) {

vertebrate\_body\_2();

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.8) {

vertebrate\_body\_2();

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.94) {

vertebrate\_body\_2();

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.875) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.93) {

vertebrate\_body\_2();

}

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.825) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.865) {

vertebrate\_body\_2();

}

}

}

}

}

}

}

}

}

}

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.3) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.37) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.19) {

vertebrate\_body\_2();

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.8) {

vertebrate\_body\_2();

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.435) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.065) {

vertebrate\_body\_2();

} else if (loc1[2] > centerpoint\_x - sizemid \* 0.185) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.125) {

vertebrate\_body\_2();

}

}

}

} else {

if (loc1[0] > centerpoint\_z + sizemax \* 0.615) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.685) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.19) {

vertebrate\_body\_2();

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.8) {

vertebrate\_body\_2();

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.06) {

vertebrate\_body\_2();

} else if (loc1[2] > centerpoint\_x - sizemid \* 0.125) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.07) {

vertebrate\_body\_2();

}

} else if (loc1[2] > centerpoint\_x - sizemid \* 0.185) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.135) {

vertebrate\_body\_2();

}

}

}

}

}

}

}

}

}

}

} else {

if (loc1[0] < centerpoint\_z + sizemax \* 0.99) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.3) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc1[2], 2))) < sizemin \* 0.0148125) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc1[2], 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc1[2], 2))) > (sizemin \* 0.0148125) \* 1.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.5025) {

vertebrate\_body\_2();

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.4975) {

vertebrate\_body\_2();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc1[2], 2))) < sizemin \* 0.0148125) {

vertebrate\_body\_2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc1[2], 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc1[2], 2))) > (sizemin \* 0.0148125) \* 1.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.5025) {

vertebrate\_body\_2();

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.4975) {

vertebrate\_body\_2();

}

}

}

}

}

}

function vertebrate\_body\_2() {

if (loc2[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.8) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.75) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.22) - loc2[2], 2))) < sizemin \* 0.02) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.17) {

make = true;

}

} else {

if (loc2[2] < centerpoint\_x - sizemid \* 0.4775018) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.5205) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.4995) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.5205) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.2495) {

make = true;

}

}

}

}

}

}

}

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

make = true;

} else if (loc2[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.02) {

make = true;

} else if (loc2[2] > centerpoint\_x - sizemid \* 0.21) {

make = true;

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.79) {

make = true;

} else if (loc2[0] > centerpoint\_z + sizemax \* 0.74) {

make = true;

} else {

if (loc2[2] < centerpoint\_x - sizemid \* 0.65) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.35) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.65) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.1) {

make = true;

} else {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.37) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.001) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.4) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.0015) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.003) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.45) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.006) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.46) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.01) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.48) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.02) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.5) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.04) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.54) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.0005) {

make = true;

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_z + sizemax \* 0.56) - loc2[0], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.00025) {

make = true;

}

}

}

}

}

}

}

}

}

} else if (loc2[2] > centerpoint\_x - sizemid \* 0.99) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.3) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.37) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.81) {

make = true;

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.8) {

make = true;

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.435) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.935) {

make = true;

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.815) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.875) {

make = true;

}

}

}

} else {

if (loc2[0] > centerpoint\_z + sizemax \* 0.615) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.685) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.81) {

make = true;

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.8) {

make = true;

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.94) {

make = true;

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.875) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.93) {

make = true;

}

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.825) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.865) {

make = true;

}

}

}

}

}

}

}

}

}

}

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.3) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.37) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.19) {

make = true;

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.8) {

make = true;

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.435) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.065) {

make = true;

} else if (loc2[2] > centerpoint\_x - sizemid \* 0.185) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.125) {

make = true;

}

}

}

} else {

if (loc2[0] > centerpoint\_z + sizemax \* 0.615) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.19) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.685) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.7) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.19) {

make = true;

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.8) {

make = true;

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.25) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.06) {

make = true;

} else if (loc2[2] > centerpoint\_x - sizemid \* 0.125) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.07) {

make = true;

}

} else if (loc2[2] > centerpoint\_x - sizemid \* 0.185) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.135) {

make = true;

}

}

}

}

}

}

}

}

}

}

} else {

if (loc2[0] < centerpoint\_z + sizemax \* 0.99) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.3) {

make = true;

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc2[2], 2))) < sizemin \* 0.0148125) {

make = true;

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc2[2], 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc2[2], 2))) > (sizemin \* 0.0148125) \* 1.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.5025) {

make = true;

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.4975) {

make = true;

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc2[2], 2))) < sizemin \* 0.0148125) {

make = true;

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc2[2], 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc2[2], 2))) > (sizemin \* 0.0148125) \* 1.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.5025) {

make = true;

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.4975) {

make = true;

}

}

}

}

}

}

function vertebrate\_connective\_1() {

if (loc1[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.79) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.76) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.05) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

vertebrate\_connective\_1b();

}

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.25) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.48) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.52) {

vertebrate\_connective\_1b();

}

}

} else {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

vertebrate\_connective\_1b();

}

}

}

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.745), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.002) {

vertebrate\_connective\_1b();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.73), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_1b();

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.75), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_1b();

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.003) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.4) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.003) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.4) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.02) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.404) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.398) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.42) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.414) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.436) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.446) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.468) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.462) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.484) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.478) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.494) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.516) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.51) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.532) {

if (loc1[0] > centerpoint\_z + sizemax \* 526) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.548) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.542) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.564) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.558) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.574) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.596) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.59) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.612) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.606) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.628) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.622) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.644) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.632) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.654) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.676) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.67) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.684) {

vertebrate\_connective\_1b();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.708) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.702) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.005) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.005) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.005) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.005) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1b();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.845), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.155), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.84), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.16), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1b();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.32), 2))) < sizemid \* 0.005) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1b();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.686), 2))) < sizemid \* 0.001) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1b();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.614), 2))) < sizemid \* 0.001) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1b();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.371), 2))) < sizemid \* 0.001) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1b();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.299), 2))) < sizemid \* 0.001) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1b();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.25), 2))) < sizemid \* 0.001) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.299) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.371) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.614) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.686) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.75), 2))) < sizemid \* 0.001) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.299) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.371) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.614) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.686) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.97), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.03), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.43), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.301) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.369) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.616) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.719) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.43), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.301) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.369) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.616) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.719) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.4), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.4), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1b();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.27) {

vertebrate\_connective\_1b();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.845), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.155), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.9), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.84), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.1), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.16), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1b();

}

}

}

}

}

function vertebrate\_connective\_1b() {

if (loc2[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.79) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.76) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.05) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.02) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

vertebrate\_connective\_1c();

}

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.25) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.48) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.52) {

vertebrate\_connective\_1c();

}

}

} else {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

vertebrate\_connective\_1c();

}

}

}

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.745), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.002) {

vertebrate\_connective\_1c();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.73), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_1c();

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.75), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_1c();

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.003) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.4) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.003) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.4) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.404) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.398) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.42) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.414) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.436) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.446) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.468) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.462) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.484) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.478) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.494) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.516) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.51) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.532) {

if (loc2[0] > centerpoint\_z + sizemax \* 526) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.548) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.542) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.564) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.558) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.574) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.596) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.59) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.612) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.606) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.628) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.622) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.644) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.632) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.654) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.676) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.67) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.684) {

vertebrate\_connective\_1c();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.708) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.702) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.005) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.005) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.005) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.005) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1c();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.845), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.155), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.33) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.43) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.84), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.16), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

vertebrate\_connective\_1c();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.32), 2))) < sizemid \* 0.005) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.225) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.3) {

vertebrate\_connective\_1c();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.686), 2))) < sizemid \* 0.001) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1c();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.614), 2))) < sizemid \* 0.001) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1c();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.371), 2))) < sizemid \* 0.001) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1c();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.299), 2))) < sizemid \* 0.001) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.35) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

vertebrate\_connective\_1c();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.65) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.25), 2))) < sizemid \* 0.001) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.299) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.371) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.614) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.686) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.75), 2))) < sizemid \* 0.001) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.299) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.371) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.614) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.686) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.97), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.03), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.43) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.453) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.43), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.301) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.369) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.616) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.719) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.43), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.301) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.369) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.616) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.719) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.4), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.4), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.312) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.358) {

vertebrate\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.627) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.673) {

vertebrate\_connective\_1c();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3031), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.3669), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.6181), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.273) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7169), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.21) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.27) {

vertebrate\_connective\_1c();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.845), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.155), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.433), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.9), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.84), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.1), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.16), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.7489), 2))) < sizemid \* 0.003) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.191) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_connective\_1c();

}

}

}

}

}

function vertebrate\_connective\_1c() {

make = true;

}

function vertebrate\_skeleton() {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.4) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.3) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.416) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.402) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.432) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.418) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.448) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.434) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.464) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.48) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.466) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.496) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.482) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.512) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.498) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.528) {

if (loc1[0] > centerpoint\_z + sizemax \* 514) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.544) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.53) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.56) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.546) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.576) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.562) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.592) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.578) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.608) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.594) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.624) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.61) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.626) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.656) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.642) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.672) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.658) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.688) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.674) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.704) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.69) {

vertebrate\_skeleton\_2();

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.706) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_2();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_2();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* (0.6 - 0.02)) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] > centerpoint\_x - sizemid \* (0.4 + 0.02)) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.96) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.92) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.04) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.02) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.96) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.04) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.02) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.02) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) > sizemid \* 0.0125) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.955) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.925) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.075) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.045) {

vertebrate\_skeleton\_2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_2();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_2();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_2();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* (0.6 - 0.02)) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] > centerpoint\_x - sizemid \* (0.4 + 0.02)) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.96) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.92) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.04) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.02) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.96) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.04) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.02) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.02) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) > sizemid \* 0.0125) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.955) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.925) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.075) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.045) {

vertebrate\_skeleton\_2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_2();

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.305) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.305) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.83) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.62) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.62) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.83) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.845), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.155), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.84), 2))) < sizemid \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.16), 2))) < sizemid \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

}

} else if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.5), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.29) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc1[2], 2))) > (sizemin \* 0.00740625) \* 1.2) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc1[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc1[2], 2))) > (sizemin \* 0.00740625) \* 1.2) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.988) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.443) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.45) {

vertebrate\_skeleton\_2();

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.98) {

vertebrate\_skeleton\_2();

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.77) {

vertebrate\_skeleton\_2();

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.5), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) > sizemid \* 0.28) {

vertebrate\_skeleton\_2();

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.83) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.85) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.31) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.4) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.45) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc1[0] + (centerpoint\_z + sizemin \* 0.84), 2))) > sizemid \* 0.005) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.83) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.85) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.4) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.45) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc1[0] + (centerpoint\_z + sizemin \* 0.84), 2))) > sizemid \* 0.005) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc1[0] + (centerpoint\_z + sizemin \* 0.84), 2))) < sizemid \* 0.004) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.315) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.685) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

vertebrate\_skeleton\_2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc1[0] + (centerpoint\_z + sizemin \* 0.84), 2))) < sizemid \* 0.006) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3135) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.317) {

vertebrate\_skeleton\_2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.683) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.687) {

vertebrate\_skeleton\_2();

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.845) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.703) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.31) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.42) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.845) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.297) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.31) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.42) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.703) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.297) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_2();

}

}

}

}

}

}

}

}

function vertebrate\_skeleton\_2() {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.025) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.4) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.3) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.416) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.402) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.432) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.418) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.448) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.434) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.464) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.48) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.466) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.496) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.482) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.512) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.498) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.528) {

if (loc2[0] > centerpoint\_z + sizemax \* 514) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.544) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.53) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.56) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.546) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.576) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.562) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.592) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.578) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.608) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.594) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.624) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.61) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.626) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.656) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.642) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.672) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.658) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.688) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.674) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.704) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.69) {

vertebrate\_skeleton\_3();

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.706) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_3();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_3();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* (0.6 - 0.02)) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] > centerpoint\_x - sizemid \* (0.4 + 0.02)) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.96) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.92) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.04) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.02) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.96) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.04) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.02) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.02) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) > sizemid \* 0.0125) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.955) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.925) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.075) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.045) {

vertebrate\_skeleton\_3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_3();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_3();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) < sizemid \* 0.025) {

vertebrate\_skeleton\_3();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* (0.6 - 0.02)) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] > centerpoint\_x - sizemid \* (0.4 + 0.02)) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.96) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.92) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.04) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.02) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.96) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.88) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.04) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.02) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.02) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) > sizemid \* 0.0125) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.955) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.925) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.075) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.045) {

vertebrate\_skeleton\_3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.94), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.06), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.01) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.715) {

vertebrate\_skeleton\_3();

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.305) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.305) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.83) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.62) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.62) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.83) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.205) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.25) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.845), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.155), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.36) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.39) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.397) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.433) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.96), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.9), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.84), 2))) < sizemid \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.04), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.1), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.225), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.16), 2))) < sizemid \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.685) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.699) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.705) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.715) {

vertebrate\_connective\_1b();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.749) {

vertebrate\_connective\_1b();

}

}

}

}

} else if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.5), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.29) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc2[2], 2))) > (sizemin \* 0.00740625) \* 1.2) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.5) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.9) - loc2[0], 2)) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc2[2], 2))) > (sizemin \* 0.00740625) \* 1.2) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.988) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.443) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.45) {

vertebrate\_skeleton\_3();

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.98) {

vertebrate\_skeleton\_3();

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.77) {

vertebrate\_skeleton\_3();

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.5), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) > sizemid \* 0.28) {

vertebrate\_skeleton\_3();

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.83) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.85) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.31) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.4) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.45) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc2[0] + (centerpoint\_z + sizemin \* 0.84), 2))) > sizemid \* 0.005) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.83) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.85) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.4) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.45) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc2[0] + (centerpoint\_z + sizemin \* 0.84), 2))) > sizemid \* 0.005) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc2[0] + (centerpoint\_z + sizemin \* 0.84), 2))) < sizemid \* 0.004) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.315) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.685) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

vertebrate\_skeleton\_3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.42), 2) + Math.pow(loc2[0] + (centerpoint\_z + sizemin \* 0.84), 2))) < sizemid \* 0.006) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3135) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.317) {

vertebrate\_skeleton\_3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.683) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.687) {

vertebrate\_skeleton\_3();

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.845) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.703) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.31) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.42) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.845) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.297) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.31) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.42) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.703) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.297) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_skeleton\_3();

}

}

}

}

}

}

}

}

function vertebrate\_skeleton\_3() {

make = true;

}

function vertebrate\_muscles() {

if (loc1[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.785) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.765) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.025) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.67) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.33) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.45) {

vertebrate\_muscle\_2();

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.55) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.3035) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.555) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.2011) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2045) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.556) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.7) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.2003) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.202) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.556) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.7) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.25) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.2057) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.204) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.584) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.25) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.676) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.686) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.614) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.624) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.676) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.686) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.614) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.624) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.361) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.371) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.299) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.309) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.361) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.371) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.299) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.309) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.694) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.24) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.921) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.24) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.921) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.816) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.874) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.936) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.064) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.126) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.184) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.816) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.874) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.936) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.064) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.126) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.184) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.961) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.876) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.929) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.826) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.865) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.0399) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.071) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.124) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.136) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.174) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.961) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.876) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.929) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.826) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.865) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.0399) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.071) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.124) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.136) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.174) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

} else {

if (loc1[2] > centerpoint\_z + sizemin \* 0.9) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.92) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.79) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.796) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.52) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.53) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.9) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.92) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.204) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.21) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.52) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.53) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.711) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.72) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.289) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.305) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.301) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.317) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.705) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.715) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.301) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.317) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.849) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.711) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.72) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.45) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.949) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.289) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.345) {

vertebrate\_muscle\_2();

}

}

}

}

}

}

}

}

function vertebrate\_muscle\_2() {

if (loc2[0] < centerpoint\_z + sizemax \* 0.3) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.785) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.765) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.025) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.67) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.33) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.45) {

vertebrate\_muscle\_3();

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.55) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.3035) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.555) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.2011) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2045) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.556) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.7) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.2003) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.202) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.556) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.7) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.25) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.2057) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.204) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.584) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.25) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.587) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.65) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.676) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.686) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.614) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.624) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.676) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.686) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.614) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.624) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.361) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.371) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.299) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.309) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.42) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.361) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.371) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.299) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.309) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.715) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.735) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.694) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.98) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.02) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.725) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.3149) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.355) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.94) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.616) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6299) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.67) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.06) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.4) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.24) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.921) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.34) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.369) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.24) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.921) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.655) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.684) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.179) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.21) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.816) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.874) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.936) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.064) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.126) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.184) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.816) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.874) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.936) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.064) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.36) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.434) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.126) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.184) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.961) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.876) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.929) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.826) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.865) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.0399) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.071) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.124) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.136) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.174) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.195) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.2) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.961) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.876) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.929) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.826) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.865) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.0101) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.0399) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.071) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.124) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.675) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.136) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.174) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.245) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.249) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

} else {

if (loc2[2] > centerpoint\_z + sizemin \* 0.9) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.92) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.79) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.796) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.52) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.53) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.9) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.92) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.204) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.21) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.52) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.53) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.711) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.72) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.289) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.318) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.305) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.301) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.317) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.97) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.987) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.705) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.715) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.301) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.317) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.849) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.711) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.72) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.45) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.835) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.949) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.289) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.345) {

vertebrate\_muscle\_3();

}

}

}

}

}

}

}

}

function vertebrate\_muscle\_3() {

make = true;

}

function vertebrate\_connective\_2() {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) > sizemid \* 0.026) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) > sizemid \* 0.026) {

vertebrate\_connective\_2b();

}

}

}

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) > sizemid \* 0.026) {

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc1[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) > sizemid \* 0.026) {

vertebrate\_connective\_2b();

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc1[1] - (centerpoint\_y - sizemin \* 0.748), 2) + Math.pow(loc1[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.003) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_2b();

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.725) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.205) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.795) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.205) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.205) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.795) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.725) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.56) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.25) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.56) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

if (loc1[2] > centerpoint\_z + sizemin \* 0.3) {

if (loc1[2] < centerpoint\_z + sizemin \* 0.31) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc1[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc1[2] > centerpoint\_y - sizemin \* 0.205) {

vertebrate\_connective\_2b();

}

}

}

}

}

}

}

}

function vertebrate\_connective\_2b() {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2))) < sizemid \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) > sizemid \* 0.026) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.335), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) > sizemid \* 0.026) {

vertebrate\_connective\_2c();

}

}

}

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2))) < sizemid \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.6 - 0.02)), 2)))) > sizemid \* 0.026) {

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.725), 2) + (Math.pow(loc2[0] - (centerpoint\_z + sizemax \* 0.65), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* (0.4 + 0.02)), 2)))) > sizemid \* 0.026) {

vertebrate\_connective\_2c();

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow(loc2[1] - (centerpoint\_y - sizemin \* 0.748), 2) + Math.pow(loc2[2] - (centerpoint\_x - sizemid \* 0.5), 2))) < sizemid \* 0.003) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.31) {

vertebrate\_connective\_2c();

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.725) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.205) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.7) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.795) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.749) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.205) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.205) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.795) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.55) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.6) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.725) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.56) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.25) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.552) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.56) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.75) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

if (loc2[2] > centerpoint\_z + sizemin \* 0.301) {

if (loc2[2] < centerpoint\_z + sizemin \* 0.31) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.201) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.799) {

if (loc2[2] < centerpoint\_y - sizemin \* 0.201) {

if (loc2[2] > centerpoint\_y - sizemin \* 0.205) {

vertebrate\_connective\_2c();

}

}

}

}

}

}

}

}

function vertebrate\_connective\_2c() {

make = true;

}

function human\_body() {

if (bvar == 2) {

human\_male\_body();

} else {

human\_female\_body();

}

}

function human\_male\_body() {

if (loc1[0] > centerpoint\_z + sizemax \* 0.001) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81379310344) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.78) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_male\_body2();

} else if (loc1[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.8) {

human\_male\_body2();

}

} else {

if (loc1[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.96) {

human\_male\_body2();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.95) {

human\_male\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.91) {

human\_male\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.84) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.87) {

human\_male\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.83) {

human\_male\_body2();

}

}

}

}

}

}

}

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.22) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body2();

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.49) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.34) - loc1[2], 2))) < sizemin \* 0.025) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.44686206896) {

human\_male\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.5) {

human\_male\_body2();

}

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.5) {

human\_male\_body2();

}

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body2();

}

}

if (loc1[1] < centerpoint\_y - sizemin \* 0.299) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.519) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.541) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.444) {

human\_male\_body2();

}

}

}

}

}

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.004) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_male\_body2();

} else if (loc1[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.2) {

human\_male\_body2();

}

} else {

if (loc1[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.04) {

human\_male\_body2();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.05) {

human\_male\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.09) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

human\_male\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.13) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.16) {

human\_male\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

human\_male\_body2();

}

}

}

}

}

}

}

}

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.25) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.9985) {

human\_male\_body2();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_male\_body2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

human\_male\_body2();

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_male\_body2();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_male\_body2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

human\_male\_body2();

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_male\_body2();

}

}

}

}

}

}

}

function human\_female\_body() {

if (loc1[0] > centerpoint\_z + sizemax \* 0.001) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81379310344) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.78) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_female\_body2();

} else if (loc1[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.8) {

human\_female\_body2();

}

} else {

if (loc1[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.96) {

human\_female\_body2();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.95) {

human\_female\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.91) {

human\_female\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.84) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.87) {

human\_female\_body2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.83) {

human\_female\_body2();

}

}

}

}

}

}

}

} else if (loc1[2] < centerpoint\_x - sizemid \* 0.22) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body2();

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.53) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.43) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.77) {

human\_female\_body2();

}

}

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.57) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.73) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.77) {

human\_female\_body2();

}

}

}

}

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.54) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.647) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.656) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.749) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.752) {

human\_female\_body2();

}

}

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.347) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.356) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.749) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.752) {

human\_female\_body2();

}

}

}

}

}

}

} else {

if (loc1[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.5) {

human\_female\_body2();

}

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.5) {

human\_female\_body2();

}

} else if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body2();

}

}

if (loc1[1] < centerpoint\_y - sizemin \* 0.299) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.519) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.541) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.444) {

human\_female\_body2();

}

}

}

}

}

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.004) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_female\_body2();

} else if (loc1[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.2) {

human\_female\_body2();

}

} else {

if (loc1[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.04) {

human\_female\_body2();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.05) {

human\_female\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.09) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.12) {

human\_female\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.13) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.16) {

human\_female\_body2();

}

}

if (loc1[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

human\_female\_body2();

}

}

}

}

}

}

}

}

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.25) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.9985) {

human\_female\_body2();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_female\_body2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

human\_female\_body2();

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_female\_body2();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_female\_body2();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc1[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

human\_female\_body2();

} else if (loc1[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_female\_body2();

}

}

}

}

}

}

}

function human\_male\_body2() {

if (loc2[0] > centerpoint\_z + sizemax \* 0.001) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81379310344) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.78) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_male\_body3();

} else if (loc2[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.8) {

human\_male\_body3();

}

} else {

if (loc2[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.96) {

human\_male\_body3();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.95) {

human\_male\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.91) {

human\_male\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.84) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.87) {

human\_male\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.83) {

human\_male\_body3();

}

}

}

}

}

}

}

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.22) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body3();

}

}

} else {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.49) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.34) - loc2[2], 2))) < sizemin \* 0.025) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.44686206896) {

human\_male\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.5) {

human\_male\_body3();

}

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.5) {

human\_male\_body3();

}

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_male\_body3();

}

}

if (loc2[1] < centerpoint\_y - sizemin \* 0.299) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.519) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.541) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.444) {

human\_male\_body3();

}

}

}

}

}

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.004) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_male\_body3();

} else if (loc2[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.2) {

human\_male\_body3();

}

} else {

if (loc2[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.04) {

human\_male\_body3();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.05) {

human\_male\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.09) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

human\_male\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.13) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.16) {

human\_male\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

human\_male\_body3();

}

}

}

}

}

}

}

}

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.25) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.9985) {

human\_male\_body3();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_male\_body3();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

human\_male\_body3();

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_male\_body3();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_male\_body3();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

human\_male\_body3();

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_male\_body3();

}

}

}

}

}

}

}

function human\_male\_body3() {

make = true;

}

function human\_female\_body2() {

if (loc2[0] > centerpoint\_z + sizemax \* 0.001) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81379310344) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.78) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_female\_body3();

} else if (loc2[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.8) {

human\_female\_body3();

}

} else {

if (loc2[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.96) {

human\_female\_body3();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.92) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.95) {

human\_female\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.91) {

human\_female\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.84) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.87) {

human\_female\_body3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.83) {

human\_female\_body3();

}

}

}

}

}

}

}

} else if (loc2[2] < centerpoint\_x - sizemid \* 0.22) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.004) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body3();

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.53) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.43) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.77) {

human\_female\_body3();

}

}

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.57) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.73) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.77) {

human\_female\_body3();

}

}

}

}

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.54) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.647) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.656) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.749) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.752) {

human\_female\_body3();

}

}

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.347) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.356) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.749) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.752) {

human\_female\_body3();

}

}

}

}

}

}

} else {

if (loc2[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.5) {

human\_female\_body3();

}

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.06551724137) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.5) {

human\_female\_body3();

}

} else if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

human\_female\_body3();

}

}

if (loc2[1] < centerpoint\_y - sizemin \* 0.299) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.519) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.541) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.444) {

human\_female\_body3();

}

}

}

}

}

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.004) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.996) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.75862068965) {

human\_female\_body3();

} else if (loc2[0] > centerpoint\_z + sizemax \* 0.48275862069) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.2) {

human\_female\_body3();

}

} else {

if (loc2[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.44827586206) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.04) {

human\_female\_body3();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.41379310344) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.05) {

human\_female\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.09) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.12) {

human\_female\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.13) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.16) {

human\_female\_body3();

}

}

if (loc2[2] > centerpoint\_x - sizemid \* 0.17) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

human\_female\_body3();

}

}

}

}

}

}

}

}

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.38) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.25) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.9985) {

human\_female\_body3();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_female\_body3();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

human\_female\_body3();

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_female\_body3();

}

}

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < sizemax \* (0.0148125 / 2)) {

human\_female\_body3();

} else if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) < (sizemax \* (0.0148125 / 2)) \* 1.5) {

if (Math.sqrt(Math.abs((Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc2[2], 2)) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2))) > (sizemax \* (0.0148125 / 2)) \* 1.3) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

human\_female\_body3();

} else if (loc2[0] < centerpoint\_z + sizemax \* 0.9308) {

human\_female\_body3();

}

}

}

}

}

}

}

function human\_female\_body3() {

make = true;

}

function human\_skeleton() {

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.03) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.52) {

human\_skeleton2();

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.54) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.524) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.56) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.544) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.564) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.6) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.584) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.62) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.604) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.64) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.624) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.644) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.68) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.664) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.7) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.684) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.72) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.704) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.74) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.724) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.76) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.744) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.78) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.764) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.923) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.784) {

human\_skeleton2();

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.519) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.47) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.53) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.555) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.4625) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.555) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.5375) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.0375) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.265) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.735) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.0355) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.50375) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.0355) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.49625) {

human\_skeleton2();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.8081) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.812) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.083) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.03) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.97) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.812) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.1) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7866666666666666) - loc1[0], 2), 2))) > sizemin \* 0.016) {

human\_skeleton2();

}

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.9) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7866666666666666) - loc1[0], 2), 2))) > sizemin \* 0.016) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.1) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7873333333333333) - loc1[0], 2), 2))) > sizemin \* 0.0135) {

human\_skeleton2();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.9) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7873333333333333) - loc1[0], 2), 2))) > sizemin \* 0.0135) {

human\_skeleton2();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.0065) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.0065) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.799) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.62) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) > sizemin \* 0.005) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.799) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.62) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) > sizemin \* 0.005) {

human\_skeleton2();

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.0035) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.0035) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.92) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.007) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.07) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.081) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.007) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.119) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.13) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.007) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.87) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.881) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc1[0], 2))) < sizemin \* 0.007) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.919) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.93) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.07) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.013) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.87) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.93) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.61) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.07) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.13) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.61) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.87) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.93) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.53) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.61) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.53) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.61) {

human\_skeleton2();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2))) < sizemin \* 0.014) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5001) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.48) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.3) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.4826666666666667) - loc1[0], 2), 2))) > sizemin \* 0.013) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.014) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5001) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.48) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.7) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.4826666666666667) - loc1[0], 2), 2))) > sizemin \* 0.013) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.7) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.483) - loc1[0], 2), 2))) > sizemin \* 0.0115) {

human\_skeleton2();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.3) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.483) - loc1[0], 2), 2))) > sizemin \* 0.0115) {

human\_skeleton2();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.006) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.483) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.47) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2))) < sizemin \* 0.006) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.483) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.47) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.471) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.31) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc1[0], 2))) > sizemin \* 0.005) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.471) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.31) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc1[0], 2))) > sizemin \* 0.005) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc1[0], 2))) < sizemin \* 0.00375) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.32) {

human\_skeleton2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.72) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc1[0], 2))) < sizemin \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.281) {

human\_skeleton2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.67) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.681) {

human\_skeleton2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.319) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.33) {

human\_skeleton2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.719) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.73) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.27) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.33) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.67) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.73) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.29) - loc1[0], 2))) > sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.33) {

human\_skeleton2();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.67) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.73) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.06) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2))) < sizemin \* 0.01) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.06) {

human\_skeleton2();

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.014) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.057) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.4) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.014) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.057) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.45) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.483) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.0526) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.01) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.2) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.483) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.0526) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.99) {

human\_skeleton2();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.98) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.4645) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.461) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.454) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.4499) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.94) - loc1[2], 2))) < sizemin \* 0.007) {

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.86) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.82) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.02) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.4645) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.461) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.454) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.4499) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.06) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.14) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.18) - loc1[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton2();

}

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton2();

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.55) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2), 2))) > sizemin \* 0.0075) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc1[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.45) - loc1[2]) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc1[0], 2), 2))) > sizemin \* 0.0075) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.24) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.99) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.997) {

human\_skeleton2();

}

} else {

if (loc1[1] > centerpoint\_y - sizemin \* 0.02) {

human\_skeleton2();

}

if (loc1[1] < centerpoint\_y - sizemin \* 0.36) {

human\_skeleton2();

}

if (loc1[0] < centerpoint\_z + sizemax \* 0.94) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

human\_skeleton2();

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.931) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.997) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) > sizemin \* 0.22) {

human\_skeleton2();

}

}

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.25) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.35) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) > sizemin \* 0.007) {

human\_skeleton2();

}

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.25) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.27) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.65) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) > sizemin \* 0.007) {

human\_skeleton2();

}

}

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.34) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.41) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.49) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.66) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) < sizemin \* 0.008) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.409) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.42) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc1[0], 2))) < sizemin \* 0.008) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.591) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.66) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.91) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

human\_skeleton2();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.34) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.91) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

human\_skeleton2();

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.335) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.345) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.26) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.655) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.665) {

human\_skeleton2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.335) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.665) {

human\_skeleton2();

}

}

}

}

}

}

}

function human\_skeleton2() {

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.03) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.52) {

human\_skeleton3();

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.54) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.524) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.56) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.544) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.564) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.6) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.584) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.62) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.604) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.64) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.624) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.644) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.68) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.664) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.7) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.684) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.72) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.704) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.74) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.724) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.76) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.744) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.78) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.764) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.923) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.784) {

human\_skeleton3();

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.519) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.47) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.53) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.555) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.4625) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.555) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.07) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.5375) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.0375) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.265) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.035) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.735) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.0355) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.50375) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.535) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.0355) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.375) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.49625) {

human\_skeleton3();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.8081) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.812) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.083) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.03) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.97) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.812) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.1) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7866666666666666) - loc2[0], 2), 2))) > sizemin \* 0.016) {

human\_skeleton3();

}

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.03) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.9) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7866666666666666) - loc2[0], 2), 2))) > sizemin \* 0.016) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.1) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7873333333333333) - loc2[0], 2), 2))) > sizemin \* 0.0135) {

human\_skeleton3();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.9) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.7873333333333333) - loc2[0], 2), 2))) > sizemin \* 0.0135) {

human\_skeleton3();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.0065) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.0065) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.79) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.799) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.62) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) > sizemin \* 0.005) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.799) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.62) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) > sizemin \* 0.005) {

human\_skeleton3();

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.0035) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.0035) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.92) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.007) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.07) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.081) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.007) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.119) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.13) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.007) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.87) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.881) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.636) - loc2[0], 2))) < sizemin \* 0.007) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.919) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.93) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.07) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.013) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.87) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.93) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.61) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.63) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.61) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.07) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.13) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.61) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.87) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.93) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.53) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.61) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.53) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.61) {

human\_skeleton3();

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2))) < sizemin \* 0.014) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5001) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.48) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.3) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.4826666666666667) - loc2[0], 2), 2))) > sizemin \* 0.013) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.014) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5001) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.48) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.7) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.4826666666666667) - loc2[0], 2), 2))) > sizemin \* 0.013) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.7) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.483) - loc2[0], 2), 2))) > sizemin \* 0.0115) {

human\_skeleton3();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.3) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.483) - loc2[0], 2), 2))) > sizemin \* 0.0115) {

human\_skeleton3();

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.006) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.483) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.47) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2))) < sizemin \* 0.006) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.483) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.47) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.471) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.31) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc2[0], 2))) > sizemin \* 0.005) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.471) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.31) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc2[0], 2))) > sizemin \* 0.005) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc2[0], 2))) < sizemin \* 0.00375) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.32) {

human\_skeleton3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.72) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.316) - loc2[0], 2))) < sizemin \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.281) {

human\_skeleton3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.67) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.681) {

human\_skeleton3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.319) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.33) {

human\_skeleton3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.719) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.73) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.27) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.33) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.67) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.73) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.316) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.29) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.29) - loc2[0], 2))) > sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.27) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.33) {

human\_skeleton3();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.67) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.73) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.06) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2))) < sizemin \* 0.01) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.29) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.06) {

human\_skeleton3();

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.014) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.057) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.4) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.014) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.057) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.45) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.483) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.0526) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.01) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.2) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.483) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.0526) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.8) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.99) {

human\_skeleton3();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.98) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.4645) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.461) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.454) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.4499) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.94) - loc2[2], 2))) < sizemin \* 0.007) {

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.86) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.82) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.02) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.4645) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.461) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.454) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.452) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.4499) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.06) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.14) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.18) - loc2[2], 2))) < sizemin \* 0.007) {

if (unused\_variable == 5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.47) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.443) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.43) {

human\_skeleton3();

}

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.426) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.419) {

human\_skeleton3();

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.55) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2), 2))) > sizemin \* 0.0075) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc2[1], 2) + Math.pow(((centerpoint\_x - sizemid \* 0.45) - loc2[2]) + Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc2[0], 2), 2))) > sizemin \* 0.0075) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.24) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.01) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.99) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.997) {

human\_skeleton3();

}

} else {

if (loc2[1] > centerpoint\_y - sizemin \* 0.03) {

human\_skeleton3();

}

if (loc2[1] < centerpoint\_y - sizemin \* 0.36) {

human\_skeleton3();

}

if (loc2[0] < centerpoint\_z + sizemax \* 0.94) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

human\_skeleton3();

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.931) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.997) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.2) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) > sizemin \* 0.22) {

human\_skeleton3();

}

}

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.25) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.35) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.4) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) > sizemin \* 0.007) {

human\_skeleton3();

}

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.25) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.27) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.65) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) > sizemin \* 0.007) {

human\_skeleton3();

}

}

}

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.34) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.41) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.49) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.66) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) < sizemin \* 0.008) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.409) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.42) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.91) - loc2[0], 2))) < sizemin \* 0.008) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.58) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.591) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.66) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.91) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

human\_skeleton3();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.26) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.34) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.91) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

human\_skeleton3();

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.335) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.345) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.26) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.655) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.665) {

human\_skeleton3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.34) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.335) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.665) {

human\_skeleton3();

}

}

}

}

}

}

}

function human\_skeleton3() {

make = true;

}

function human\_connective\_2() {

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.881) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.0835) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.086) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.47) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.53) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.49) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.52) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.86) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.56) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.785) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.86) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.76) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.05) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.45) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.05) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.24) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.76) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.45) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.7) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.56) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.206) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.213) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.491) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.509) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.48586206896) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.79) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.69) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.36) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.221) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.25) {

human\_connective\_2b();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.69) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.16) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.779) {

human\_connective\_2b();

}

}

}

}

}

}

}

function human\_connective\_2b() {

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.881) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.0835) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.086) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.47) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.53) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.49) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.52) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.02) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.86) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.56) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.785) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.86) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.76) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.05) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.45) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.05) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.24) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.03) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.76) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.55) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.66) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.45) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.7) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.8) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.56) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.909) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.206) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.213) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.491) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.509) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.47586206896) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.48586206896) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.371) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.79) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.69) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.36) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.221) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.25) {

human\_connective\_2c();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.69) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.692) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.16) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.75) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.779) {

human\_connective\_2c();

}

}

}

}

}

}

}

function human\_connective\_2c() {

make = true;

}

function human\_muscles() {

if (loc1[0] > centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) {

if (loc1[0] < centerpoint\_z + sizemax \* ((centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) + 0.01)) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.373) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.46) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) {

if (loc1[0] < centerpoint\_z + sizemax \* ((centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) + 0.01)) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.373) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.54) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.56) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.905) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.29) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.336) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.344) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.905) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.29) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.656) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.664) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.9107) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.9125) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.241) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.356) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.481) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.519) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.91) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.9106) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.241) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.299) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.481) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.519) {

human\_muscle\_2();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.47186206896) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.67) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.372) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.377) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.68) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.77) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.37) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.68) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.77) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.377) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.6901) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.718) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.355) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.245) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.26) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.717) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.719) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.355) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.245) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.5) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.717) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.719) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.155) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.49) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.761) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.6901) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.718) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.155) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.76) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.769) {

human\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.095) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.105) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.895) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.905) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.92) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.05) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.15) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.05) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.15) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.85) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.95) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.85) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.95) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.11) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.89) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.91) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.89) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.91) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.11) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.09) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.11) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.09) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.11) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.199) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.0101) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.199) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.0101) {

human\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.961) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.949) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.921) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.909) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.881) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.869) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.841) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.829) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.039) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.051) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.079) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.091) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.119) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.131) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.159) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.171) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.199) {

human\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.961) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.949) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.921) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.909) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.881) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.869) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.841) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.829) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.039) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.051) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.079) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.091) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.119) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.131) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.159) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.171) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.199) {

human\_muscle\_2();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.2869) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3131) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.32) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.6869) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.7131) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.72) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.32) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.4) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.32) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.8) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.4) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.8) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.31) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.31) {

human\_muscle\_2();

}

}

}

}

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.71) {

human\_muscle\_2();

}

}

}

}

}

}

}

}

function human\_muscle\_2() {

if (loc2[0] > centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) {

if (loc2[0] < centerpoint\_z + sizemax \* ((centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) + 0.01)) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.373) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.44) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.46) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) {

if (loc2[0] < centerpoint\_z + sizemax \* ((centerpoint\_z + sizemax \* (0.93103448275 + (0.0148125 / 2) \* 1.4)) + 0.01)) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.373) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.378) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.54) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.56) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.905) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.29) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.336) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.344) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.905) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.934) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.29) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.656) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.664) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.9107) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.9125) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.241) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.356) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.481) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.519) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.91) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.9106) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.241) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.299) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.481) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.519) {

human\_muscle\_3();

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.47186206896) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.67) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.372) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.377) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.68) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.77) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.37) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.372) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.68) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.77) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.377) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.23) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.77) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.6901) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.718) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.355) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.245) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.26) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.717) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.719) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.355) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.245) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.5) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.717) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.719) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.155) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.49) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.761) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.6901) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.718) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.09) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.155) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.76) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.769) {

human\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.095) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.105) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.895) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.76) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.81) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.04) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.06) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.905) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.92) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.05) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.15) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.05) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.15) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.85) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.95) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.58) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.75) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.85) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.95) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.11) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.89) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.91) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.89) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.91) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.1) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.11) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.09) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.11) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.57) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.03) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.09) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.11) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.199) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.0101) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.4828) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.199) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.0101) {

human\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.961) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.949) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.921) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.909) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.881) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.869) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.841) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.829) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.039) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.051) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.079) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.091) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.119) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.131) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.159) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.345) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.171) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.199) {

human\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.989) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.961) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.949) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.921) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.909) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.881) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.869) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.841) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.829) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.801) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.011) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.039) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.051) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.079) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.091) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.119) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.131) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.159) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.415) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.49) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.375) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.171) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.199) {

human\_muscle\_3();

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.2869) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3131) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.32) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.6869) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.46) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.36) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.37) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.7131) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.72) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.32) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.4) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.32) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.8) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.2) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.4) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.27) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.44) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.6) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.8) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.31) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.33) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.29) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.31) {

human\_muscle\_3();

}

}

}

}

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.035) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.2) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.3751) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.69) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.71) {

human\_muscle\_3();

}

}

}

}

}

}

}

}

function human\_muscle\_3() {

make = true;

}

function human\_connective\_1() {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.03) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.004) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.88) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc1[2], 2))) < sizemin \* 0.02) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.518) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.526) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.538) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.546) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.558) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.566) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.578) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.586) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.598) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.606) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.618) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.626) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.638) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.646) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.658) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.666) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.678) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.686) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.698) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.706) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.718) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.726) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.738) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.746) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.758) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.766) {

human\_connective\_1b();

}

}

if (loc1[0] > centerpoint\_z + sizemax \* 0.778) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.786) {

human\_connective\_1b();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.526) - loc1[0], 2))) < sizemin \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.52) - loc1[0], 2))) < sizemin \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.514) - loc1[0], 2))) < sizemin \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.5083) - loc1[0], 2))) < sizemin \* 0.01) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.503) - loc1[0], 2))) < sizemin \* 0.007) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.533) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.529) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.525) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.521) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.517) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.513) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.509) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.505) - loc1[0], 2))) < sizemin \* 0.005) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.502) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.1) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.025) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.1) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.025) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.55) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.55) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1b();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.98) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.94) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.86) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.82) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.02) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.45) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.06) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.14) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.18) - loc1[2], 2))) < sizemin \* 0.005) {

if (loc1[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc1[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1b();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.46) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.32) {

human\_connective\_1b();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.72) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.44) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.44) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.27) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.27) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.2) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.2) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc1[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.76) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.12) {

human\_connective\_1b();

}

}

if (loc1[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc1[2] > centerpoint\_x - sizemid \* 0.92) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.75) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.75) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.58) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.58) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.57) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.11) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.57) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.11) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.025) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.45) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.975) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.45) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.935) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.895) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.855) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.815) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.065) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.105) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.145) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.185) - loc1[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc1[0], 2))) < sizemin \* 0.003) {

if (loc1[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc1[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1b();

}

}

}

}

function human\_connective\_1b() {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.03) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.004) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.88) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc2[2], 2))) < sizemin \* 0.02) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.518) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.526) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.538) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.546) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.558) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.566) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.578) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.586) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.598) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.606) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.618) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.626) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.638) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.646) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.658) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.666) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.678) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.686) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.698) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.706) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.718) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.726) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.738) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.746) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.758) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.766) {

human\_connective\_1c();

}

}

if (loc2[0] > centerpoint\_z + sizemax \* 0.778) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.786) {

human\_connective\_1c();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.526) - loc2[0], 2))) < sizemin \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.52) - loc2[0], 2))) < sizemin \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.514) - loc2[0], 2))) < sizemin \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.5083) - loc2[0], 2))) < sizemin \* 0.01) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.055) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.503) - loc2[0], 2))) < sizemin \* 0.007) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.533) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.529) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.525) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.521) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.517) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.513) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.509) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.505) - loc2[0], 2))) < sizemin \* 0.005) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.502) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.3) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.7) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.1) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.025) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.1) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.025) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.55) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.55) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.49) {

human\_connective\_1c();

}

}

}

if (unused\_variable == 5) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.98) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.94) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.86) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.82) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.02) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.45) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.06) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.14) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.36) - loc2[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.18) - loc2[2], 2))) < sizemin \* 0.005) {

if (loc2[0] < centerpoint\_z + sizemax \* 0.5) {

if (loc2[0] > centerpoint\_z + sizemax \* 0.42) {

human\_connective\_1c();

}

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.365) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.46) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.28) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.32) {

human\_connective\_1c();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.68) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.72) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.44) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.44) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.27) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.27) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.2) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.3) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.2) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.7) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.2) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.32) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin \* 0.05) - loc2[1], 2) + Math.pow((centerpoint\_z + sizemax \* 0.76) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[2] < centerpoint\_x - sizemid \* 0.08) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.12) {

human\_connective\_1c();

}

}

if (loc2[2] < centerpoint\_x - sizemid \* 0.88) {

if (loc2[2] > centerpoint\_x - sizemid \* 0.92) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.75) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.75) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.58) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.58) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.3) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.1) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.57) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.11) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.9) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.57) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.021) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.11) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.025) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.45) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.975) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.45) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.935) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.895) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.855) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.815) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.065) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.105) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.145) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_x - sizemid \* 0.185) - loc2[2], 2) + Math.pow((centerpoint\_z + sizemax \* 0.42) - loc2[0], 2))) < sizemin \* 0.003) {

if (loc2[1] < centerpoint\_y - sizemin \* 0.341) {

if (loc2[1] > centerpoint\_y - sizemin \* 0.379) {

human\_connective\_1c();

}

}

}

}

function human\_connective\_1c() {

make = true;

}

function invertebrate\_male() {

unused\_variable = 5;

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 2;

sizemin = sizemid;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1597 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1597 = 0; count1597 < repeat\_end1597; count1597++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

fly\_brain();

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.2, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.1, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.2]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.2, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.8, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.9]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.1, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.2]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.8, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.9]);

list\_tissues.push(['geno', centerpoint\_z + sizemax \* 0.875, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.7, centerpoint\_x - sizemid \* 0.7]);

list\_tissues.push(['duct1', centerpoint\_z + sizemax \* 0.9000000000000001, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.55, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.6]);

list\_tissues.push(['visceralx', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemid \* 0.6]);

list\_tissues.push(['nephridium', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.60000000000001, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.15, centerpoint\_x - sizemid \* 0.7]);

list\_tissues.push(['liver', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.8, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemid \* 0.35]);

list\_tissues.push(['marrow', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.6]);

exempt = [null];

if (unused\_variable == 5) {

invertebrate\_2();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30;

cedey = 1.2375855e+30;

cedez = 1.237585532e+31;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1598 = 0; count1598 < repeat\_x; count1598++) {

for (var count1599 = 0; count1599 < repeat\_y; count1599++) {

for (var count1600 = 0; count1600 < repeat\_z; count1600++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = sizemid \* 0.01;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['heart'];

layer();

if (make == true) {

cardiomyocyte\_1();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30;

cedey = 1.2375855e+30;

cedez = 1.237585532e+31;

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 1.2375855e+30 / 2;

spawnery = centerpoint\_y - 1.2375855e+30 / 2;

spawnerz = centerpoint\_z + 1.237585532e+31 / 2;

}

repeat\_x = Math.round(sizemid / cedex - 1);

repeat\_y = Math.round(sizemin / cedey - 1);

repeat\_z = Math.round(sizemax / cedez - 1);

for (var count1601 = 0; count1601 < repeat\_x; count1601++) {

for (var count1602 = 0; count1602 < repeat\_y; count1602++) {

for (var count1603 = 0; count1603 < repeat\_z; count1603++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = sizemid \* 0.01;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['heart'];

layer();

if (make == true) {

cardiomyocyte\_1();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

innervate\_1();

}

exempt = [null];

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1604 = 0; count1604 < repeat\_x; count1604++) {

for (var count1605 = 0; count1605 < repeat\_y; count1605++) {

for (var count1606 = 0; count1606 < repeat\_z; count1606++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

endocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1607 = 0; count1607 < repeat\_x; count1607++) {

for (var count1608 = 0; count1608 < repeat\_y; count1608++) {

for (var count1609 = 0; count1609 < repeat\_z; count1609++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_molecules();

screen\_cells();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

nephrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1610 = 0; count1610 < repeat\_x; count1610++) {

for (var count1611 = 0; count1611 < repeat\_y; count1611++) {

for (var count1612 = 0; count1612 < repeat\_z; count1612++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['liver'];

find2();

if (make == true) {

hepatocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1613 = 0; count1613 < repeat\_x; count1613++) {

for (var count1614 = 0; count1614 < repeat\_y; count1614++) {

for (var count1615 = 0; count1615 < repeat\_z; count1615++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['marrow'];

find2();

if (make == true) {

hematocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1616 = 0; count1616 < repeat\_x; count1616++) {

for (var count1617 = 0; count1617 < repeat\_y; count1617++) {

for (var count1618 = 0; count1618 < repeat\_z; count1618++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['geno'];

find2();

if (make == true) {

genocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.1, centerpoint\_y - sizemin \* 0.8, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.15, centerpoint\_y - sizemin \* 0.9, centerpoint\_x - sizemid \* 0.99]);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.901;

spawnery = centerpoint\_y - sizemin \* 0.8;

for (var count1619 = 0; count1619 < 5; count1619++) {

for (var count1620 = 0; count1620 < 5; count1620++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.1500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.45, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.4500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.450000001, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

for (var count1621 = 0; count1621 < 5; count1621++) {

for (var count1622 = 0; count1622 < 5; count1622++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.4500000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.64, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.6400000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.640000001, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

for (var count1623 = 0; count1623 < 5; count1623++) {

for (var count1624 = 0; count1624 < 5; count1624++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.6400000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.71, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7100000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.710000001, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

for (var count1625 = 0; count1625 < 5; count1625++) {

for (var count1626 = 0; count1626 < 5; count1626++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.710000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.79, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.790000001, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

for (var count1627 = 0; count1627 < 5; count1627++) {

for (var count1628 = 0; count1628 < 5; count1628++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.790000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.85, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.850000001, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

for (var count1629 = 0; count1629 < 5; count1629++) {

for (var count1630 = 0; count1630 < 5; count1630++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.850000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.89, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.890000001, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

spawnery = spawnery - sizemin \* 2e-7;

}

spawnery = spawnery + sizemin \* 0.000001;

spawnerx = spawnerx - sizemid \* 2e-7;

}

spawnerx = spawnerx + sizemid \* 0.000001;

spawnery = spawnery - sizemin \* 0.000002;

}

spawnery = spawnery + sizemin \* 0.00001;

spawnerx = spawnerx - sizemid \* 0.000002;

}

spawnerx = spawnerx + sizemid \* 0.00001;

spawnery = spawnery - sizemin \* 0.00002;

}

spawnery = spawnery + sizemin \* 0.0001;

spawnerx = spawnerx - sizemid \* 0.00002;

}

spawnerx = spawnerx + sizemid \* 0.0001;

spawnery = spawnery - sizemin \* 0.0002;

}

spawnery = spawnery + sizemin \* 0.001;

spawnerx = spawnerx - sizemid \* 0.0002;

}

spawnerx = spawnerx + sizemid \* 0.001;

spawnery = spawnery - sizemin \* 0.002;

}

spawnery = spawnery + sizemin \* 0.01;

spawnerx = spawnerx - sizemid \* 0.002;

}

spawnerx = spawnerx + sizemid \* 0.01;

spawnery = spawnery - sizemin \* 0.02;

}

spawnery = spawnery + sizemin \* 0.1;

spawnerx = spawnerx - sizemid \* 0.02;

}

}

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.1, centerpoint\_y - sizemin \* 0.8, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.15, centerpoint\_y - sizemin \* 0.9, centerpoint\_x - sizemid \* 0.01]);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.009;

spawnery = centerpoint\_y - sizemin \* 0.8;

for (var count1631 = 0; count1631 < 5; count1631++) {

for (var count1632 = 0; count1632 < 5; count1632++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.1500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.45, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.4500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.450000001, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

for (var count1633 = 0; count1633 < 5; count1633++) {

for (var count1634 = 0; count1634 < 5; count1634++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.4500000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.64, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.6400000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.640000001, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

for (var count1635 = 0; count1635 < 5; count1635++) {

for (var count1636 = 0; count1636 < 5; count1636++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.6400000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.71, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7100000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.710000001, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

for (var count1637 = 0; count1637 < 5; count1637++) {

for (var count1638 = 0; count1638 < 5; count1638++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.710000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.79, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.790000001, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

for (var count1639 = 0; count1639 < 5; count1639++) {

for (var count1640 = 0; count1640 < 5; count1640++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.790000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.85, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.850000001, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

for (var count1641 = 0; count1641 < 5; count1641++) {

for (var count1642 = 0; count1642 < 5; count1642++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.850000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.89, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.890000001, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

spawnery = spawnery - sizemin \* 2e-7;

}

spawnery = spawnery + sizemin \* 0.000001;

spawnerx = spawnerx - sizemid \* 2e-7;

}

spawnerx = spawnerx + sizemid \* 0.000001;

spawnery = spawnery - sizemin \* 0.000002;

}

spawnery = spawnery + sizemin \* 0.00001;

spawnerx = spawnerx - sizemid \* 0.000002;

}

spawnerx = spawnerx + sizemid \* 0.00001;

spawnery = spawnery - sizemin \* 0.00002;

}

spawnery = spawnery + sizemin \* 0.0001;

spawnerx = spawnerx - sizemid \* 0.00002;

}

spawnerx = spawnerx + sizemid \* 0.0001;

spawnery = spawnery - sizemin \* 0.0002;

}

spawnery = spawnery + sizemin \* 0.001;

spawnerx = spawnerx - sizemid \* 0.0002;

}

spawnerx = spawnerx + sizemid \* 0.001;

spawnery = spawnery - sizemin \* 0.002;

}

spawnery = spawnery + sizemin \* 0.01;

spawnerx = spawnerx - sizemid \* 0.002;

}

spawnerx = spawnerx + sizemid \* 0.01;

spawnery = spawnery - sizemin \* 0.02;

}

spawnery = spawnery + sizemin \* 0.1;

spawnerx = spawnerx - sizemid \* 0.02;

}

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x;

spawnerz = centerpoint\_z;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

list\_tissues.push(['spiracle', spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x;

spawnerz = centerpoint\_z;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

loc1 = [spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0];

loc2 = [spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05];

spiracle();

}

spawnerz = centerpoint\_z;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x - sizemid \* 0.95;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

list\_tissues.push(['spiracle1', spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

}

spawnerz = centerpoint\_z;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x - sizemid \* 0.95;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

loc1 = [spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0];

loc2 = [spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05];

spiracle();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1643 = 0; count1643 < generation\_parameter\_x; count1643++) {

for (var count1644 = 0; count1644 < generation\_parameter\_3; count1644++) {

for (var count1645 = 0; count1645 < generation\_parameter\_z; count1645++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = ['leg'];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

exempt = [null];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == false) {

make = true;

collagen();

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 150) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) <= sizemin / 2) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) >= sizemin \* (0.5 - 1 / 1000)) {

make = true;

collagen();

}

}

} else if (loc1[0] >= centerpoint\_z + sizemax \* 0.95) {

if (loc1[0] <= centerpoint\_z + sizemax \* 0.96) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) <= sizemin / 2) {

make = true;

collagen();

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

}

}

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) >= sizemin \* (0.5 - 1 / 1000)) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

make = false;

thickabs = 1.887317842e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.856378206e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['geno'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.021008013e+29;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.05194765e+28;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

} else {

invertebrate\_endoskeleton();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1646 = 0; count1646 < generation\_parameter\_x; count1646++) {

for (var count1647 = 0; count1647 < generation\_parameter\_3; count1647++) {

for (var count1648 = 0; count1648 < generation\_parameter\_z; count1648++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = ['leg'];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

exempt = [null];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == false) {

make = true;

collagen();

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 150) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) >= sizemin \* (0.5 - 1 / 1000)) {

make = true;

collagen();

}

}

} else if (loc1[0] >= centerpoint\_z + sizemax \* 0.95) {

if (loc1[0] <= centerpoint\_z + sizemax \* 0.96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

collagen();

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

}

}

} else if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) >= sizemin \* (0.5 - 1 / 1000)) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

make = false;

thickabs = 1.887317842e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.856378206e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['geno'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.021008013e+29;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.05194765e+28;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

} else {

invertebrate\_endoskeleton();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 4.950341882e+29;

cedey = 2.475170941e+29;

cedez = 4.950341882e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1649 = 0; count1649 < repeat\_x; count1649++) {

for (var count1650 = 0; count1650 < repeat\_y; count1650++) {

for (var count1651 = 0; count1651 < repeat\_z; count1651++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 4.95034188e+29, spawnery - 2.47517094e+29, spawnerx - 4.95034188e+29];

make = true;

exempt = ['heart', 'cap'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin \* 0.45) {

make = true;

erythrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.423223291e+26 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 9.28189103e+25;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1652 = 0; count1652 < generation\_parameter\_x; count1652++) {

for (var count1653 = 0; count1653 < generation\_parameter\_3; count1653++) {

for (var count1654 = 0; count1654 < generation\_parameter\_z; count1654++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 9.281891e+25, cursor1y - 1.54698183e+26, cursor1x - 1.42322329e+26];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = 3.712756412e+26;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle', 'filament'];

layer();

if (make == true) {

randomize\_lipid\_1();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

fluids\_1();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 2.115113925e+26 \* 1;

density\_parameter\_y = 2.115113925e+26 \* 1;

density\_parameter\_z = 2.115113925e+26;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1655 = 0; count1655 < generation\_parameter\_x; count1655++) {

for (var count1656 = 0; count1656 < generation\_parameter\_3; count1656++) {

for (var count1657 = 0; count1657 < generation\_parameter\_z; count1657++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

make = true;

loc2 = [cursor1z + 2.11511392e+26, cursor1y - 2.11511392e+26, cursor1x - 2.11511392e+26];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = false;

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament', 'spiracle', 'visceralx', 'nephridium', 'duct1'];

find2();

if (make == true) {

randomize\_air\_molecule();

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.103963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == false) {

if (unused\_variable == 5) {

make = false;

thickabs = 2.497170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1001;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == false) {

randomize\_air\_molecule();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament', 'spiracle', 'visceralx', 'nephridium', 'duct1'];

find2();

if (make == true) {

randomize\_air\_molecule();

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function invertebrate\_female() {

unused\_variable = 5;

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 2;

sizemin = sizemid;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1658 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1658 = 0; count1658 < repeat\_end1658; count1658++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

fly\_brain();

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.2, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.1, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.2]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.2, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.8, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.9]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.1, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.2]);

list\_tissues.push(['leg', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.0015, centerpoint\_x - sizemid \* 0.8, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.9]);

list\_tissues.push(['ovary', centerpoint\_z + sizemax \* 0.875, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.75, centerpoint\_x - sizemid \* 0.75]);

list\_tissues.push(['geno', centerpoint\_z + sizemax \* 0.9000000000000001, centerpoint\_y - sizemin \* 0.47, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.50000000000001, centerpoint\_x - sizemid \* 0.50000000000001]);

list\_tissues.push(['duct1', centerpoint\_z + sizemax \* 0.9000000000000001, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.963, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemid \* 0.65]);

list\_tissues.push(['visceralx', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemid \* 0.6]);

list\_tissues.push(['nephridium', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.60000000000001, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.15, centerpoint\_x - sizemid \* 0.7]);

list\_tissues.push(['liver', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.8, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemid \* 0.35]);

list\_tissues.push(['marrow', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.6]);

exempt = [null];

if (unused\_variable == 5) {

invertebrate\_2();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30;

cedey = 1.2375855e+30;

cedez = 1.237585532e+31;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1659 = 0; count1659 < repeat\_x; count1659++) {

for (var count1660 = 0; count1660 < repeat\_y; count1660++) {

for (var count1661 = 0; count1661 < repeat\_z; count1661++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = sizemid \* 0.01;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['heart'];

layer();

if (make == true) {

cardiomyocyte\_1();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30;

cedey = 1.2375855e+30;

cedez = 1.237585532e+31;

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 1.2375855e+30 / 2;

spawnery = centerpoint\_y - 1.2375855e+30 / 2;

spawnerz = centerpoint\_z + 1.237585532e+31 / 2;

}

repeat\_x = Math.round(sizemid / cedex - 1);

repeat\_y = Math.round(sizemin / cedey - 1);

repeat\_z = Math.round(sizemax / cedez - 1);

for (var count1662 = 0; count1662 < repeat\_x; count1662++) {

for (var count1663 = 0; count1663 < repeat\_y; count1663++) {

for (var count1664 = 0; count1664 < repeat\_z; count1664++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 6.187926734e+29];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = sizemid \* 0.01;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['heart'];

layer();

if (make == true) {

cardiomyocyte\_1();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

innervate\_1();

}

exempt = [null];

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1665 = 0; count1665 < repeat\_x; count1665++) {

for (var count1665b = 0; count1665b < repeat\_y; count1665b++) {

for (var count1667 = 0; count1667 < repeat\_z; count1667++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['geno'];

find2();

if (make == true) {

genocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1668 = 0; count1668 < repeat\_x; count1668++) {

for (var count1669 = 0; count1669 < repeat\_y; count1669++) {

for (var count1670 = 0; count1670 < repeat\_z; count1670++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

endocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1671 = 0; count1671 < repeat\_x; count1671++) {

for (var count1672 = 0; count1672 < repeat\_y; count1672++) {

for (var count1673 = 0; count1673 < repeat\_z; count1673++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_molecules();

screen\_cells();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

nephrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1674 = 0; count1674 < repeat\_x; count1674++) {

for (var count1675 = 0; count1675 < repeat\_y; count1675++) {

for (var count1676 = 0; count1676 < repeat\_z; count1676++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['liver'];

find2();

if (make == true) {

hepatocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1677 = 0; count1677 < repeat\_x; count1677++) {

for (var count1678 = 0; count1678 < repeat\_y; count1678++) {

for (var count1679 = 0; count1679 < repeat\_z; count1679++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['marrow'];

find2();

if (make == true) {

hematocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1680 = 0; count1680 < repeat\_x; count1680++) {

for (var count1681 = 0; count1681 < repeat\_y; count1681++) {

for (var count1682 = 0; count1682 < repeat\_z; count1682++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

if (make == true) {

make = false;

target2 = null;

target1 = ['ovary'];

find2();

if (make == true) {

ovocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1683 = 0; count1683 < repeat\_x; count1683++) {

for (var count1684 = 0; count1684 < repeat\_y; count1684++) {

for (var count1685 = 0; count1685 < repeat\_z; count1685++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

melanocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.1, centerpoint\_y - sizemin \* 0.8, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.15, centerpoint\_y - sizemin \* 0.9, centerpoint\_x - sizemid \* 0.99]);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.901;

spawnery = centerpoint\_y - sizemin \* 0.8;

for (var count1686 = 0; count1686 < 5; count1686++) {

for (var count1687 = 0; count1687 < 5; count1687++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.1500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.45, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.4500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.450000001, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

for (var count1688 = 0; count1688 < 5; count1688++) {

for (var count1689 = 0; count1689 < 5; count1689++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.4500000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.64, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.6400000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.640000001, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

for (var count1690 = 0; count1690 < 5; count1690++) {

for (var count1691 = 0; count1691 < 5; count1691++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.6400000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.71, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7100000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.710000001, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

for (var count1692 = 0; count1692 < 5; count1692++) {

for (var count1693 = 0; count1693 < 5; count1693++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.710000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.79, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.790000001, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

for (var count1694 = 0; count1694 < 5; count1694++) {

for (var count1695 = 0; count1695 < 5; count1695++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.790000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.85, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.850000001, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

for (var count1696 = 0; count1696 < 5; count1696++) {

for (var count1697 = 0; count1697 < 5; count1697++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.850000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.89, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.890000001, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

spawnery = spawnery - sizemin \* 2e-7;

}

spawnery = spawnery + sizemin \* 0.000001;

spawnerx = spawnerx - sizemid \* 2e-7;

}

spawnerx = spawnerx + sizemid \* 0.000001;

spawnery = spawnery - sizemin \* 0.000002;

}

spawnery = spawnery + sizemin \* 0.00001;

spawnerx = spawnerx - sizemid \* 0.000002;

}

spawnerx = spawnerx + sizemid \* 0.00001;

spawnery = spawnery - sizemin \* 0.00002;

}

spawnery = spawnery + sizemin \* 0.0001;

spawnerx = spawnerx - sizemid \* 0.00002;

}

spawnerx = spawnerx + sizemid \* 0.0001;

spawnery = spawnery - sizemin \* 0.0002;

}

spawnery = spawnery + sizemin \* 0.001;

spawnerx = spawnerx - sizemid \* 0.0002;

}

spawnerx = spawnerx + sizemid \* 0.001;

spawnery = spawnery - sizemin \* 0.002;

}

spawnery = spawnery + sizemin \* 0.01;

spawnerx = spawnerx - sizemid \* 0.002;

}

spawnerx = spawnerx + sizemid \* 0.01;

spawnery = spawnery - sizemin \* 0.02;

}

spawnery = spawnery + sizemin \* 0.1;

spawnerx = spawnerx - sizemid \* 0.02;

}

}

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.1, centerpoint\_y - sizemin \* 0.8, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.15, centerpoint\_y - sizemin \* 0.9, centerpoint\_x - sizemid \* 0.01]);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.009;

spawnery = centerpoint\_y - sizemin \* 0.8;

for (var count1698 = 0; count1698 < 5; count1698++) {

for (var count1699 = 0; count1699 < 5; count1699++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.1500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.45, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.4500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.450000001, spawnery - sizemin \* 0.01, spawnerx - sizemid \* 0.01]);

for (var count1700 = 0; count1700 < 5; count1700++) {

for (var count1701 = 0; count1701 < 5; count1701++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.4500000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.64, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.6400000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.640000001, spawnery - sizemin \* 0.001, spawnerx - sizemid \* 0.001]);

for (var count1702 = 0; count1702 < 5; count1702++) {

for (var count1703 = 0; count1703 < 5; count1703++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.6400000010000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.71, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7100000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.710000001, spawnery - sizemin \* 0.0001, spawnerx - sizemid \* 0.0001]);

for (var count1704 = 0; count1704 < 5; count1704++) {

for (var count1705 = 0; count1705 < 5; count1705++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.710000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.79, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.7900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.790000001, spawnery - sizemin \* 0.00001, spawnerx - sizemid \* 0.00001]);

for (var count1706 = 0; count1706 < 5; count1706++) {

for (var count1707 = 0; count1707 < 5; count1707++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.790000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.85, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8500000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.850000001, spawnery - sizemin \* 0.000001, spawnerx - sizemid \* 0.000001]);

for (var count1708 = 0; count1708 < 5; count1708++) {

for (var count1709 = 0; count1709 < 5; count1709++) {

list\_tissues.push(['filament', centerpoint\_z + sizemax \* 0.850000001000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.89, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

list\_tissues.push(['gill', centerpoint\_z + sizemax \* 0.8900000000000001, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, centerpoint\_z + sizemax \* 0.890000001, spawnery - sizemin \* 1e-7, spawnerx - sizemid \* 1e-7]);

spawnery = spawnery - sizemin \* 2e-7;

}

spawnery = spawnery + sizemin \* 0.000001;

spawnerx = spawnerx - sizemid \* 2e-7;

}

spawnerx = spawnerx + sizemid \* 0.000001;

spawnery = spawnery - sizemin \* 0.000002;

}

spawnery = spawnery + sizemin \* 0.00001;

spawnerx = spawnerx - sizemid \* 0.000002;

}

spawnerx = spawnerx + sizemid \* 0.00001;

spawnery = spawnery - sizemin \* 0.00002;

}

spawnery = spawnery + sizemin \* 0.0001;

spawnerx = spawnerx - sizemid \* 0.00002;

}

spawnerx = spawnerx + sizemid \* 0.0001;

spawnery = spawnery - sizemin \* 0.0002;

}

spawnery = spawnery + sizemin \* 0.001;

spawnerx = spawnerx - sizemid \* 0.0002;

}

spawnerx = spawnerx + sizemid \* 0.001;

spawnery = spawnery - sizemin \* 0.002;

}

spawnery = spawnery + sizemin \* 0.01;

spawnerx = spawnerx - sizemid \* 0.002;

}

spawnerx = spawnerx + sizemid \* 0.01;

spawnery = spawnery - sizemin \* 0.02;

}

spawnery = spawnery + sizemin \* 0.1;

spawnerx = spawnerx - sizemid \* 0.02;

}

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x;

spawnerz = centerpoint\_z;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

list\_tissues.push(['spiracle', spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x;

spawnerz = centerpoint\_z;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

loc1 = [spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0];

loc2 = [spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05];

spiracle();

}

spawnerz = centerpoint\_z;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x - sizemid \* 0.95;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

list\_tissues.push(['spiracle1', spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0, spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

list\_tissues.push(['spiracle', 0, 0, 0, 10, 10, 10]);

}

spawnerz = centerpoint\_z;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = centerpoint\_x - sizemid \* 0.95;

while (spawnerz < (centerpoint\_z + sizemax) \* 0.85) {

spawnerz = (spawnerz + sizemax) \* 0.1;

loc1 = [spawnerz + sizemax \* 0, spawnery - sizemin \* 0, spawnerx - sizemid \* 0];

loc2 = [spawnerz + sizemin \* 0.05, spawnery - sizemin \* 0.05, spawnerx - sizemin \* 0.05];

spiracle();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1710 = 0; count1710 < generation\_parameter\_x; count1710++) {

for (var count1711 = 0; count1711 < generation\_parameter\_3; count1711++) {

for (var count1712 = 0; count1712 < generation\_parameter\_z; count1712++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = ['leg'];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

exempt = [null];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == false) {

make = true;

collagen();

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 150) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) <= sizemin / 2) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) >= sizemin \* (0.5 - 1 / 1000)) {

make = true;

collagen();

}

}

} else if (loc1[0] >= centerpoint\_z + sizemax \* 0.95) {

if (loc1[0] <= centerpoint\_z + sizemax \* 0.96) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2)) + Math.abs(Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2)))) <= sizemin / 2) {

make = true;

collagen();

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

}

}

} else if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) >= sizemin \* (0.5 - 1 / 1000)) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) <= sizemin / 2) {

make = true;

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

make = false;

thickabs = 1.887317842e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30 - 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['geno'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.021008013e+29;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.05194765e+28;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ovary'];

layer();

if (make == true) {

collagen();

} else {

invertebrate\_endoskeleton();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1713 = 0; count1713 < generation\_parameter\_x; count1713++) {

for (var count1714 = 0; count1714 < generation\_parameter\_3; count1714++) {

for (var count1715 = 0; count1715 < generation\_parameter\_z; count1715++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = ['leg'];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

exempt = [null];

make = true;

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == false) {

make = true;

collagen();

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 150) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) <= sizemin / 2) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) >= sizemin \* (0.5 - 1 / 1000)) {

make = true;

collagen();

}

}

} else if (loc1[0] >= centerpoint\_z + sizemax \* 0.95) {

if (loc1[0] <= centerpoint\_z + sizemax \* 0.96) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) <= sizemin / 2) {

make = true;

collagen();

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

}

}

} else if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) >= sizemin \* (0.5 - 1 / 1000)) {

if (Math.sqrt(Math.abs(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2)) + Math.abs(Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2)))) <= sizemin / 2) {

make = true;

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

make = false;

thickabs = 1.887317842e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceralx'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30 - 1.856378206e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['geno'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.980136753e+30;

thickabs2 = 1.856378206e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.021008013e+29;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.05194765e+28;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

collagen();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 1.021008013e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ovary'];

layer();

if (make == true) {

collagen();

} else {

invertebrate\_endoskeleton();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 4.950341882e+29;

cedey = 2.475170941e+29;

cedez = 4.950341882e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1716 = 0; count1716 < repeat\_x; count1716++) {

for (var count1717 = 0; count1717 < repeat\_y; count1717++) {

for (var count1718 = 0; count1718 < repeat\_z; count1718++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 4.950341881e+29, spawnery - 2.47517094e+29, spawnerx - 4.950341881e+29];

make = true;

exempt = ['heart', 'cap'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

if (Math.sqrt(Math.abs(Math.pow(Math.abs((centerpoint\_y - sizemin / 2) - loc1[1]), 2) + Math.pow(Math.abs((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0]), 2))) <= sizemin \* 0.45) {

make = true;

erythrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.423223291e+26 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 9.28189103e+25;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1719 = 0; count1719 < generation\_parameter\_x; count1719++) {

for (var count1720 = 0; count1720 < generation\_parameter\_3; count1720++) {

for (var count1721 = 0; count1721 < generation\_parameter\_z; count1721++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 9.281891e+25, cursor1y - 1.54698183e+26, cursor1x - 1.42322329e+26];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

thickabs = 3.712756412e+26;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['spiracle', 'filament'];

layer();

if (make == true) {

randomize\_lipid\_1();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

fluids\_1();

air\_female\_1();

}

function invertebrate\_2() {

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.8 - 0);

var repeat\_end1722 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1722 = 0; count1722 < repeat\_end1722; count1722++) {

var repeat\_end1723 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1723 = 0; count1723 < repeat\_end1723; count1723++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.2 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.1 - 0);

var repeat\_end1724 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1724 = 0; count1724 < repeat\_end1724; count1724++) {

var repeat\_end1725 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1725 = 0; count1725 < repeat\_end1725; count1725++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.2 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.29 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.8 - 0);

var repeat\_end1726 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1726 = 0; count1726 < repeat\_end1726; count1726++) {

var repeat\_end1727 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1727 = 0; count1727 < repeat\_end1727; count1727++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.4 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.29 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.29 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.1 - 0);

var repeat\_end1728 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1728 = 0; count1728 < repeat\_end1728; count1728++) {

var repeat\_end1729 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1729 = 0; count1729 < repeat\_end1729; count1729++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.4 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.29 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.49 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.8 - 0);

var repeat\_end1730 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1730 = 0; count1730 < repeat\_end1730; count1730++) {

var repeat\_end1731 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1731 = 0; count1731 < repeat\_end1731; count1731++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.6 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.49 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.49 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.1 - 0);

var repeat\_end1732 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1732 = 0; count1732 < repeat\_end1732; count1732++) {

var repeat\_end1733 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1733 = 0; count1733 < repeat\_end1733; count1733++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.6 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.49 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.69 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.8 - 0);

var repeat\_end1734 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1734 = 0; count1734 < repeat\_end1734; count1734++) {

var repeat\_end1735 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1735 = 0; count1735 < repeat\_end1735; count1735++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.8 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.69 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.69 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.2 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.1 - 0);

var repeat\_end1736 = Math.floor((sizemin / 10) / 2.475170941e+30);

for (var count1736 = 0; count1736 < repeat\_end1736; count1736++) {

var repeat\_end1737 = Math.floor((sizemid / 10) / 2.475170941e+30);

for (var count1737 = 0; count1737 < repeat\_end1737; count1737++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.8 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.69 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.05 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.4 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.85 - 0);

var repeat\_end1738 = Math.floor((sizemin \* 0.09) / 2.475170941e+30);

for (var count1738 = 0; count1738 < repeat\_end1738; count1738++) {

var repeat\_end1739 = Math.floor((sizemid \* 0.085) / 2.475170941e+30);

for (var count1739 = 0; count1739 < repeat\_end1739; count1739++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.85 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.05 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0.05 - 0);

spawnery = centerpoint\_y - (sizemin \* 0.4 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0.05 - 0);

var repeat\_end1740 = Math.floor((sizemin \* 0.09) / 2.475170941e+30);

for (var count1740 = 0; count1740 < repeat\_end1740; count1740++) {

var repeat\_end1741 = Math.floor((sizemid \* 0.085) / 2.475170941e+30);

for (var count1741 = 0; count1741 < repeat\_end1741; count1741++) {

tendon\_1();

spawnerz = spawnerz + 1.23758547e+30;

while (spawnerz < centerpoint\_z + (sizemax \* 0.85 - 0)) {

myocyte();

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.05 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.4;

spawnery = centerpoint\_y - sizemin \* 0.95;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

glial\_cell\_1();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

cardioblast();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

myoblast();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

smooth\_muscle\_cell();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

epithelioblast();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

melanoblast();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

melanocyte();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

alpha\_cell();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

beta\_cell();

}

spawnerx = spawnerx - 6.187927353e+29;

spawnerz = centerpoint\_z + sizemin \* 0.1;

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

spawnerz = spawnerz + 6.187927353e+29;

fibroblast();

}

}

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + sizemin \* 0.9;

spawnery = centerpoint\_y - sizemin \* 0.82;

spawnerx = centerpoint\_x - sizemid \* 0.45;

while (spawnerz >= centerpoint\_z + sizemax \* 0.45) {

list\_tissues.push(['heart', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0, centerpoint\_z - sizemax \* 0.13, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.1]);

list\_tissues.push(['heart', centerpoint\_z - sizemax \* 0.1300000000001, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0.04, centerpoint\_z - sizemax \* 0.1499999999999, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.06]);

list\_tissues.push(['heart', centerpoint\_z - sizemax \* 0.135, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0, centerpoint\_z - sizemax \* 0.1499999999999, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.03]);

list\_tissues.push(['heart', centerpoint\_z - sizemax \* 0.135, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0.07, centerpoint\_z - sizemax \* 0.1499999999999, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.1]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.10000000000001, centerpoint\_x - sizemid \* 0.055, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.045]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y + sizemin \* 1e-12, centerpoint\_x - sizemid \* 0.055, centerpoint\_z - sizemax \* 0.07, centerpoint\_y + sizemin \* 0.01, centerpoint\_x - sizemid \* 0.045]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.045, centerpoint\_x + sizemid \* 1e-12, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.055, centerpoint\_x + sizemid \* 0.01]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.045, centerpoint\_x - sizemid \* 0.10000000000001, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.055, centerpoint\_x + sizemid \* 0.11]);

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1x = spawnerx;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1x > spawnerx - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y + sizemin \* 1e-12, centerpoint\_x - 6.187927353e+31, cursor1z + 6.187927353e+31, centerpoint\_y + sizemin \* 0.01, centerpoint\_x - 6.187927353e+31]);

cursor1x = cursor1x - 3.093963676e+32;

}

cursor1x = spawnerx;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1x = spawnerx;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1x > spawnerx - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - sizemin \* 0.10000000000001, centerpoint\_x - 6.187927353e+31, cursor1z + 6.187927353e+31, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - 6.187927353e+31]);

cursor1x = cursor1x - 3.093963676e+32;

}

cursor1x = spawnerx;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1y = spawnery;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1y > spawnery - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 1e-12, cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 0.01]);

cursor1y = cursor1y - 3.093963676e+32;

}

cursor1y = spawnery;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1y = spawnery;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1y > spawnery - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x - sizemid \* 0.10000000000001, cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 0.11]);

cursor1y = cursor1y - 3.093963676e+32;

}

cursor1y = spawnery;

cursor1z = cursor1z + 3.093963676e+32;

}

}

spawnerz = spawnerz - sizemax \* 0.15;

}

list\_tissues.push(['heart', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0, centerpoint\_z - sizemax \* 0.13, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.1]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.1300000000001, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemid \* 0.04, centerpoint\_z - sizemax \* 0.1499999999999, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.06]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.10000000000001, centerpoint\_x - sizemid \* 0.055, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.045]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y + sizemin \* 1e-12, centerpoint\_x - sizemid \* 0.055, centerpoint\_z - sizemax \* 0.07, centerpoint\_y + sizemin \* 0.01, centerpoint\_x - sizemid \* 0.045]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.045, centerpoint\_x + sizemid \* 1e-12, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.055, centerpoint\_x + sizemid \* 0.01]);

list\_tissues.push(['cap', centerpoint\_z - sizemax \* 0.06, centerpoint\_y - sizemin \* 0.045, centerpoint\_x - sizemid \* 0.10000000000001, centerpoint\_z - sizemax \* 0.07, centerpoint\_y - sizemin \* 0.055, centerpoint\_x + sizemid \* 0.11]);

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1x = spawnerx;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1x > spawnerx - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y + sizemin \* 1e-12, centerpoint\_x - 6.187927353e+31, cursor1z + 6.187927353e+31, centerpoint\_y + sizemin \* 0.01, centerpoint\_x - 6.187927353e+31]);

cursor1x = cursor1x - 3.093963676e+32;

}

cursor1x = spawnerx;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1x = spawnerx;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1x > spawnerx - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - sizemin \* 0.10000000000001, centerpoint\_x - 6.187927353e+31, cursor1z + 6.187927353e+31, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - 6.187927353e+31]);

cursor1x = cursor1x - 3.093963676e+32;

}

cursor1x = spawnerx;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1y = spawnery;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1y > spawnery - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 1e-12, cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 0.01]);

cursor1y = cursor1y - 3.093963676e+32;

}

cursor1y = spawnery;

cursor1z = cursor1z + 3.093963676e+32;

}

}

if (sizemax > 1.23758547e+34) {

cursor1z = spawnerz;

cursor1y = spawnery;

while (cursor1z < spawnerz + sizemax \* 0.1) {

while (cursor1y > spawnery - sizemax \* 0.1) {

list\_tissues.push(['cap', cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x - sizemid \* 0.10000000000001, cursor1z + 6.187927353e+31, centerpoint\_y - 6.187927353e+31, centerpoint\_x + sizemid \* 0.11]);

cursor1y = cursor1y - 3.093963676e+32;

}

cursor1y = spawnerx;

cursor1z = cursor1z + 3.093963676e+32;

}

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.35;

spawnery = centerpoint\_y - sizemin \* 0.95;

spawnerz = centerpoint\_z + sizemin \* 0.05;

while (spawnerz < centerpoint\_z + sizemax \* 0.3) {

spawnerz = spawnerz + 6.187927353e+29;

antennocyte();

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0;

spawnery = centerpoint\_y - sizemin \* 0;

spawnerz = centerpoint\_z + sizemin \* 0.05;

while (spawnerx > centerpoint\_x - sizemid \* 0.1) {

while (spawnery > centerpoint\_y + sizemin \* 0.1) {

proophalmocyte();

spawnery = spawnery - 6.187927353e+29;

}

spawnery = centerpoint\_y - sizemin \* 0;

spawnerx = spawnerx - 6.187927353e+29;

}

}

}

function fluids\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1742 = 0; count1742 < generation\_parameter\_x; count1742++) {

for (var count1743 = 0; count1743 < generation\_parameter\_3; count1743++) {

for (var count1744 = 0; count1744 < generation\_parameter\_z; count1744++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

glucose();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

glucose();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

glucose();

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

glucose();

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

glucose();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1745 = 0; count1745 < generation\_parameter\_x; count1745++) {

for (var count1746 = 0; count1746 < generation\_parameter\_3; count1746++) {

for (var count1747 = 0; count1747 < generation\_parameter\_z; count1747++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_fat();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

randomize\_fat();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

randomize\_fat();

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_fat();

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

randomize\_fat();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1748 = 0; count1748 < generation\_parameter\_x; count1748++) {

for (var count1749 = 0; count1749 < generation\_parameter\_3; count1749++) {

for (var count1750 = 0; count1750 < generation\_parameter\_z; count1750++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_amino\_acid();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_amino\_acid();

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

randomize\_amino\_acid();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1751 = 0; count1751 < generation\_parameter\_x; count1751++) {

for (var count1752 = 0; count1752 < generation\_parameter\_3; count1752++) {

for (var count1753 = 0; count1753 < generation\_parameter\_z; count1753++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_ribonucleotide();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_ribonucleotide();

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

randomize\_ribonucleotide();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1754 = 0; count1754 < generation\_parameter\_x; count1754++) {

for (var count1755 = 0; count1755 < generation\_parameter\_3; count1755++) {

for (var count1756 = 0; count1756 < generation\_parameter\_z; count1756++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_vitamin();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

randomize\_vitamin();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

randomize\_vitamin();

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomize\_vitamin();

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

randomize\_vitamin();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1757 = 0; count1757 < generation\_parameter\_x; count1757++) {

for (var count1758 = 0; count1758 < generation\_parameter\_3; count1758++) {

for (var count1759 = 0; count1759 < generation\_parameter\_z; count1759++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

set\_earth\_rotation();

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1760 = 0; count1760 < generation\_parameter\_x; count1760++) {

for (var count1761 = 0; count1761 < generation\_parameter\_3; count1761++) {

for (var count1762 = 0; count1762 < generation\_parameter\_z; count1762++) {

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

set\_earth\_rotation();

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['heart', 'cap', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

if ((loc1[2] < brainz + 7.425512824e+30 && loc1[2] > brainz) && ((loc1[1] > brainy - 9.900683765e+30 && loc1[1] < brainy) && (loc1[0] > brainx - 1.732619659e+31 && loc1[0] < brainx))) {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == true) {

if ((loc1[2] < brainz + 7.425512824e+30 && loc1[2] > brainz) && ((loc1[1] > brainy - 9.900683765e+30 && loc1[1] < brainy) && (loc1[0] > brainx - 1.732619659e+31 && loc1[0] < brainx))) {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 2.475170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == true) {

if ((loc1[2] < brainz + 7.425512824e+30 && loc1[2] > brainz) && ((loc1[1] > brainy - 9.900683765e+30 && loc1[1] < brainy) && (loc1[0] > brainx - 1.732619659e+31 && loc1[0] < brainx))) {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

make = true;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

exempt = ['gill', 'filament', 'leg'];

screen\_cells();

screen\_molecules();

screen\_tissues();

if (make == true) {

if ((loc1[2] < brainz + 7.425512824e+30 && loc1[2] > brainz) && ((loc1[1] > brainy - 9.900683765e+30 && loc1[1] < brainy) && (loc1[0] > brainx - 1.732619659e+31 && loc1[0] < brainx))) {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

if ((loc1[2] < brainz + 7.425512824e+30 && loc1[2] > brainz) && ((loc1[1] > brainy - 9.900683765e+30 && loc1[1] < brainy) && (loc1[0] > brainx - 1.732619659e+31 && loc1[0] < brainx))) {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function air\_female\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 2.115113925e+26 \* 1;

density\_parameter\_y = 2.115113925e+26 \* 1;

density\_parameter\_z = 2.115113925e+26;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1763 = 0; count1763 < generation\_parameter\_x; count1763++) {

for (var count1764 = 0; count1764 < generation\_parameter\_3; count1764++) {

for (var count1765 = 0; count1765 < generation\_parameter\_z; count1765++) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.11511392e+26, cursor1y - 2.11511392e+26, cursor1x - 2.11511392e+26];

if (unused\_variable == 5) {

if (make == false) {

make = true;

} else {

if (loc1[0] <= centerpoint\_z + sizemax / 96) {

if (Math.sqrt(Math.abs(Math.pow((centerpoint\_y - sizemin / 2) - loc1[1], 2) + Math.pow((centerpoint\_x - sizemid / 2) - loc1.slice(-1)[0], 2))) <= sizemin / 2) {

make = false;

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament', 'spiracle', 'visceralx', 'nephridium', 'duct1'];

find2();

if (make == true) {

randomize\_air\_molecule();

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.103963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament'];

layer();

if (make == false) {

if (unused\_variable == 5) {

make = false;

thickabs = 2.497170941e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['gill'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

} else {

if (unused\_variable == 5) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1001;

thickrel2 = 0;

target2 = null;

target1 = ['duct1'];

layer();

if (make == false) {

randomize\_air\_molecule();

} else {

if (unused\_variable == 5) {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['filament', 'spiracle', 'visceralx', 'nephridium', 'duct1'];

find2();

if (make == true) {

randomize\_air\_molecule();

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function gland\_1() {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.315, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.901, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.501]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.313, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.901, centerpoint\_y - sizemin \* 0.315, centerpoint\_x - sizemid \* 0.501]);

box = [centerpoint\_z + sizemax \* 0.88, centerpoint\_y - sizemin \* 0.311, centerpoint\_x - sizemid \* 0.45, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.319, centerpoint\_x - sizemid \* 0.55];

loc1 = [centerpoint\_z + sizemax \* 0.88, centerpoint\_y - sizemin \* 0.311, centerpoint\_z + sizemax \* 0.93];

loc1 = [centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.319, centerpoint\_x - sizemid \* 0.55];

duct();

}

function vertebrate\_male() {

unused\_variable = 5;

bvar = 1;

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 3;

sizemin = sizemid;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1766 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1766 = 0; count1766 < repeat\_end1766; count1766++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + sizemax \* (0.55 / 35))]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + sizemax \* (0.55 / 35))]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4)]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

if (unused\_variable == 5) {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - 0.305]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - 0.305]);

box = [centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemid \* 0.299, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.4001, centerpoint\_x - sizemid \* 0.5];

loc1 = [centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemid \* 0.3];

loc2 = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - 0.305];

ductsize = 6.187927353e+30;

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - 0.305]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.35]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.47, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.355]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.43, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.47, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - 0.38]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.4]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.41]);

box = [centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

loc1 = [centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.4];

loc2 = [centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.41];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.34, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36]);

box = [centerpoint\_z + sizemax \* 0.32, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

loc1 = [centerpoint\_z + sizemax \* 0.34, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355];

loc2 = [centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36];

duct();

}

}

if (unused\_variable == 5) {

tissues\_list.push(['bladder', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.32]);

tissues\_list.push(['duct1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.225]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.23]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.23, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.24]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.529, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.53, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.549, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.569, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.57, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.25]);

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.51;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.529) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.51;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.53;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.549) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.53;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.55;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.569) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.55;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.57;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.6) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.57;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnerx > centerpoint\_x - sizemin \* 0.5) {

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.35, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnerx > centerpoint\_x - sizemin \* 0.52) {

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.35, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.5]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.349;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.5]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.26;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.349;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.33, centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.32, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.33, centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.335]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.315, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.225, centerpoint\_z + sizemax \* 0.32, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.33]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.4995, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.55];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.47];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.335, centerpoint\_z + sizemax \* 0.42573, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.38];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345];

duct();

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.2;

spawnery = centerpoint\_y - sizemin \* 0.795;

spawnerz = centerpoint\_z + sizemax \* 0.3;

while (spawnerz < centerpoint\_z + sizemax \* 0.75) {

while (spawnerx > centerpoint\_x - sizemid \* 0.8) {

tissues\_list.push(['sweat', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 3, centerpoint\_y - sizemin \* 1, spawnerx - 6.187927353e+29 \* 3]);

spawnerx = spawnerx - 6.187927353e+29 \* 45;

}

spawnerx = centerpoint\_x - sizemid \* 0.2;

spawnerz = spawnerz + 6.187927353e+29 \* 45;

}

}

if (unused\_variable == 5) {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.37) + (sizemid \* 0.001) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.001) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.37) + (sizemid \* 0.001) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.001) / 2]);

box = [centerpoint\_z + sizemax \* 0.369, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.375, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.4) + (sizemid \* 0.0015) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0015) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.4) + (sizemid \* 0.0015) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0015) / 2]);

box = [centerpoint\_z + sizemax \* 0.399, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.41, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.42, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.42) + (sizemid \* 0.003) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.003) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.42, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.42) + (sizemid \* 0.003) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.003) / 2]);

box = [centerpoint\_z + sizemax \* 0.419, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.435, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.45) + (sizemid \* 0.006) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.006) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.45) + (sizemid \* 0.006) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.006) / 2]);

box = [centerpoint\_z + sizemax \* 0.439, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.455, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.46) + (sizemid \* 0.01) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.01) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.46) + (sizemid \* 0.01) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.01) / 2]);

box = [centerpoint\_z + sizemax \* 0.459, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.475, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.48) + (sizemid \* 0.02) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.02) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.48) + (sizemid \* 0.02) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.02) / 2]);

box = [centerpoint\_z + sizemax \* 0.479, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.5) + (sizemid \* 0.04) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.04) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.5) + (sizemid \* 0.04) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.04) / 2]);

box = [centerpoint\_z + sizemax \* 0.499, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.535, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.54) + (sizemid \* 0.0005) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0005) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.54) + (sizemid \* 0.0005) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0005) / 2]);

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.545, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.56, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.56) + (sizemid \* 0.00025) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.00025) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.56, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.56) + (sizemid \* 0.00025) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.00025) / 2]);

box = [centerpoint\_z + sizemax \* 0.559, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.565, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

}

if (unused\_variable == 5) {

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.2, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.8]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.38]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.62, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.46, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.52]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.96, centerpoint\_y - sizemin \* 0.46, centerpoint\_x - sizemid \* 0.52]);

}

gland\_1();

musculature\_1();

vertebrate\_cells1();

vertebrate\_brain();

innervate\_2();

vertebrate\_circulatory();

vertebrate\_cells2();

fluids\_2();

air\_2();

}

function vertebrate\_female() {

unused\_variable = 5;

bvar = 1;

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 3;

sizemin = sizemid;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_end1767 = string\_solar\_distance\_\_length\_parameter - 1;

for (var count1767 = 0; count1767 < repeat\_end1767; count1767++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + sizemax \* (0.55 / 35))]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + sizemax \* (0.55 / 35))]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 12)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 11)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 10)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 9)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 8)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 7)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5), centerpoint\_z + sizemax \* 0.365, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 6)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 5)]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4)]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 1), centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 4), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 2), centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3), centerpoint\_x - sizemid \* (0.3 + (sizemax \* (0.55 / 35)) \* 3)]);

if (unused\_variable == 5) {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - 0.305]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - 0.305]);

box = [centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemid \* 0.299, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.4001, centerpoint\_x - sizemid \* 0.5];

loc1 = [centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemid \* 0.3];

loc2 = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - 0.305];

ductsize = 6.187927353e+30;

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* (0.22 + sizemax \* (0.55 / 35)), centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - 0.305]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.35]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.47, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.355]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.43, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.47, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - 0.38]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.4]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.41]);

box = [centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

loc1 = [centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.4];

loc2 = [centerpoint\_z + sizemax \* 0.485, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.41];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.34, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36]);

box = [centerpoint\_z + sizemax \* 0.32, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

loc1 = [centerpoint\_z + sizemax \* 0.34, centerpoint\_y - sizemin \* 0.41, centerpoint\_x - sizemid \* 0.355];

loc2 = [centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.415, centerpoint\_x - 0.36];

duct();

}

}

if (unused\_variable == 5) {

tissues\_list.push(['bladder', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.32]);

tissues\_list.push(['duct1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.225]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.23]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.23, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.24]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.529, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.53, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.549, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.569, centerpoint\_x - sizemid \* 0.25]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.57, centerpoint\_x - sizemid \* 0.24, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.25]);

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.51;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.529) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.51;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.53;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.549) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.53;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.55;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.569) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.55;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.24;

spawnery = centerpoint\_y - sizemin \* 0.57;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerz < centerpoint\_z + sizemax \* 0.55) {

while (spawnery > centerpoint\_y - sizemin \* 0.6) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, centerpoint\_x - sizemid \* 0.3]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.57;

spawnerz = spawnerz + 6.187927353e+29 \* 6;

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnerx > centerpoint\_x - sizemin \* 0.5) {

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.35, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnerx > centerpoint\_x - sizemin \* 0.52) {

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.35, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.5;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.25;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.5]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.48;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.349;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.5]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.26;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.349;

while (spawnery > centerpoint\_y - sizemin \* 0.52) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.33, centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.51, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.36, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.33, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.335]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.525, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.335, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.65]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.525, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.645, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.59, centerpoint\_x - sizemid \* 0.65]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.4995, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.55];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.5];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.505];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.47];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.405];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345]);

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.335, centerpoint\_z + sizemax \* 0.42573, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.38];

loc1 = [centerpoint\_z + sizemax \* 0.38, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.34];

loc2 = [centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.505, centerpoint\_x - sizemid \* 0.345];

duct();

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.2;

spawnery = centerpoint\_y - sizemin \* 0.795;

spawnerz = centerpoint\_z + sizemax \* 0.3;

while (spawnerz < centerpoint\_z + sizemax \* 0.75) {

while (spawnerx > centerpoint\_x - sizemid \* 0.8) {

tissues\_list.push(['sweat', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 3, centerpoint\_y - sizemin \* 1, spawnerx - 6.187927353e+29 \* 3]);

spawnerx = spawnerx - 6.187927353e+29 \* 45;

}

spawnerx = centerpoint\_x - sizemid \* 0.2;

spawnerz = spawnerz + 6.187927353e+29 \* 45;

}

}

if (unused\_variable == 5) {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.37) + (sizemid \* 0.001) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.001) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.37, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.37) + (sizemid \* 0.001) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.001) / 2]);

box = [centerpoint\_z + sizemax \* 0.369, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.375, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.4) + (sizemid \* 0.0015) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0015) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.4) + (sizemid \* 0.0015) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0015) / 2]);

box = [centerpoint\_z + sizemax \* 0.399, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.41, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.42, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.42) + (sizemid \* 0.003) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.003) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.42, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.42) + (sizemid \* 0.003) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.003) / 2]);

box = [centerpoint\_z + sizemax \* 0.419, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.435, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.45) + (sizemid \* 0.006) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.006) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.45) + (sizemid \* 0.006) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.006) / 2]);

box = [centerpoint\_z + sizemax \* 0.439, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.455, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.46) + (sizemid \* 0.01) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.01) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.46) + (sizemid \* 0.01) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.01) / 2]);

box = [centerpoint\_z + sizemax \* 0.459, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.475, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.48) + (sizemid \* 0.02) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.02) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.48) + (sizemid \* 0.02) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.02) / 2]);

box = [centerpoint\_z + sizemax \* 0.479, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.495, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.5) + (sizemid \* 0.04) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.04) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.5) + (sizemid \* 0.04) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.04) / 2]);

box = [centerpoint\_z + sizemax \* 0.499, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.535, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.54) + (sizemid \* 0.0005) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0005) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.54) + (sizemid \* 0.0005) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.0005) / 2]);

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.545, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.56, centerpoint\_y - sizemin \* 0.03, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.56) + (sizemid \* 0.00025) / 2, centerpoint\_y - sizemin \* 0.11, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.00025) / 2]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.56, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.7, (centerpoint\_z + sizemax \* 0.56) + (sizemid \* 0.00025) / 2, centerpoint\_y - sizemin \* 0.13, (centerpoint\_x - sizemid \* 0.7) - (sizemid \* 0.00025) / 2]);

box = [centerpoint\_z + sizemax \* 0.559, centerpoint\_y - sizemin \* 0.105, centerpoint\_x - sizemid \* 0.66, centerpoint\_z + sizemax \* 0.565, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.74];

}

if (unused\_variable == 5) {

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.2, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.8]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.38]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.62, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.38, centerpoint\_x - sizemid \* 0.32, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.68]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.46, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.52]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.44, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.96, centerpoint\_y - sizemin \* 0.46, centerpoint\_x - sizemid \* 0.52]);

}

if (unused\_variable == 5) {

tissues\_list.push(['vhomol', centerpoint\_z + sizemax \* 0.1, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemid \* 0.7]);

tissues\_list.push(['uterus', centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.595, centerpoint\_x - sizemid \* 0.645, centerpoint\_z + sizemax \* 0.41, centerpoint\_y - sizemin \* 0.605, centerpoint\_x - sizemid \* 0.655]);

tissues\_list.push(['uterus', centerpoint\_z + sizemax \* 0.41, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemid \* 0.7]);

tissues\_list.push(['isthmus', centerpoint\_z + sizemax \* 0.46, centerpoint\_y - sizemin \* 0.59, centerpoint\_x - sizemid \* 0.64, centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.61, centerpoint\_x - sizemid \* 0.66]);

tissues\_list.push(['magnum', centerpoint\_z + sizemax \* 0.48, centerpoint\_y - sizemin \* 0.59, centerpoint\_x - sizemid \* 0.64, centerpoint\_z + sizemax \* 0.525, centerpoint\_y - sizemin \* 0.61, centerpoint\_x - sizemid \* 0.66]);

tissues\_list.push(['infundibulum', centerpoint\_z + sizemax \* 0.525, centerpoint\_y - sizemin \* 0.59, centerpoint\_x - sizemid \* 0.64, centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.61, centerpoint\_x - sizemid \* 0.66]);

}

gland\_1();

musculature\_1();

vertebrate\_cells1();

vertebrate\_brain();

innervate\_2();

vertebrate\_circulatory();

ovary\_1();

vertebrate\_cells2();

fluids\_2();

air\_2();

}

function musculature\_1() {

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0 - 0);

spawnery = centerpoint\_y - (sizemin \* 0 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0 - 0);

muscle = false;

var repeat\_end1768 = Math.floor(sizemin / 2.475170941e+30);

for (var count1768 = 0; count1768 < repeat\_end1768; count1768++) {

var repeat\_end1769 = Math.floor(sizemid / 2.475170941e+30);

for (var count1769 = 0; count1769 < repeat\_end1769; count1769++) {

while (spawnerz < centerpoint\_z + (sizemax \* 1 - 0)) {

if (muscle == true) {

make = true;

exempt = [null];

loc1 = [spawnerz + 1.23758547e+30 \* 1, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 2, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_muscle();

if (make == true) {

myocyte();

} else {

muscle = false;

tendon\_2();

}

} else {

muscle = false;

tendon\_2();

}

} else {

make = true;

exempt = [null];

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 1, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_muscle();

if (make == true) {

loc1 = [spawnerz + 1.23758547e+30 \* 1, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 2, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_muscle();

if (make == true) {

tendon\_1();

muscle = true;

}

}

}

}

}

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

}

function vertebrate\_cells1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1770 = 0; count1770 < repeat\_x; count1770++) {

for (var count1771 = 0; count1771 < repeat\_y; count1771++) {

for (var count1772 = 0; count1772 < repeat\_z; count1772++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['vhomol'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

vagocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['uterus'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

uterocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['isthmus'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

isthmocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['magnum'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

magnocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['infundibulum'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.589, centerpoint\_x - sizemid \* 0.639, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.611, centerpoint\_x - sizemid \* 0.661];

box3();

if (make == false) {

infundibulocyte();

}

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductC'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

ductocyte();

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1773 = 0; count1773 < repeat\_x; count1773++) {

for (var count1774 = 0; count1774 < repeat\_y; count1774++) {

for (var count1775 = 0; count1775 < repeat\_z; count1775++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

endocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1776 = 0; count1776 < repeat\_x; count1776++) {

for (var count1777 = 0; count1777 < repeat\_y; count1777++) {

for (var count1778 = 0; count1778 < repeat\_z; count1778++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

nephrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1779 = 0; count1779 < repeat\_x; count1779++) {

for (var count1780 = 0; count1780 < repeat\_y; count1780++) {

for (var count1781 = 0; count1781 < repeat\_z; count1781++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'bladder'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

urocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1782 = 0; count1782 < repeat\_x; count1782++) {

for (var count1783 = 0; count1783 < repeat\_y; count1783++) {

for (var count1784 = 0; count1784 < repeat\_z; count1784++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

if (loc1[1] >= centerpoint\_y - sizemin \* 0.2) {

mammocyte();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1785 = 0; count1785 < repeat\_x; count1785++) {

for (var count1786 = 0; count1786 < repeat\_y; count1786++) {

for (var count1787 = 0; count1787 < repeat\_z; count1787++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductA'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

sustenocyte();

} else {

genocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1788 = 0; count1788 < repeat\_x; count1788++) {

for (var count1789 = 0; count1789 < repeat\_y; count1789++) {

for (var count1790 = 0; count1790 < repeat\_z; count1790++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductB'];

layer();

if (make == true) {

epycyte();

}

}

spawnerz = spawnery + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1791 = 0; count1791 < generation\_parameter\_x; count1791++) {

for (var count1792 = 0; count1792 < generation\_parameter\_3; count1792++) {

for (var count1793 = 0; count1793 < generation\_parameter\_z; count1793++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 3.279601497e+28 \* 2.5;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'sweat', 'ductA', 'ductB', 'ductC', 'nephridium', 'bladder', 'ureter', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.589, centerpoint\_x - sizemid \* 0.639, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.611, centerpoint\_x - sizemid \* 0.661];

box3();

if (make == false) {

collagen();

}

}

}

}

}

} else {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (var count1794 = 0; count1794 < generation\_parameter\_x; count1794++) {

for (var count1795 = 0; count1795 < generation\_parameter\_3; count1795++) {

for (var count1796 = 0; count1796 < generation\_parameter\_z; count1796++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (loc1[1] < centerpoint\_y - sizemin \* 0.2) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 3.279601497e+28 \* 2.5;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'sweat', 'ductA', 'ductB', 'ductC', 'nephridium', 'bladder', 'ureter', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.589, centerpoint\_x - sizemid \* 0.639, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.611, centerpoint\_x - sizemid \* 0.661];

box3();

if (make == false) {

collagen();

}

}

}

}

}

} else {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function ovary\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1797 = 0; count1797 < repeat\_x; count1797++) {

for (var count1798 = 0; count1798 < repeat\_y; count1798++) {

for (var count1799 = 0; count1799 < repeat\_z; count1799++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = 1;

thickbod2 = 0;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['infundibulum', null, null];

box = [centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.59, centerpoint\_x - sizemid \* 0.64, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.61, centerpoint\_x - sizemid \* 0.66];

box3();

if (make == true) {

make = false;

layer();

if (make == true) {

ovocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

}

function vertebrate\_cells2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1800 = 0; count1800 < repeat\_x; count1800++) {

for (var count1801 = 0; count1801 < repeat\_y; count1801++) {

for (var count1802 = 0; count1802 < repeat\_z; count1802++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.4;

thickrel2 = 0;

target2 = null;

target1 = ['uterus'];

layer();

if (make == true) {

smooth\_muscle\_cell();

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['vhomol', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

smooth\_muscle\_cell();

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0.02;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3', 'visceral4'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1803 = 0; count1803 < repeat\_x; count1803++) {

for (var count1804 = 0; count1804 < repeat\_y; count1804++) {

for (var count1805 = 0; count1805 < repeat\_z; count1805++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.4995, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.55];

box3();

if (make == true) {

vesicocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1806 = 0; count1806 < repeat\_x; count1806++) {

for (var count1807 = 0; count1807 < repeat\_y; count1807++) {

for (var count1808 = 0; count1808 < repeat\_z; count1808++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.45, centerpoint\_y - sizemin \* 0.6, centerpoint\_x - sizemid \* 0.47];

box3();

if (make == true) {

procyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1809 = 0; count1809 < repeat\_x; count1809++) {

for (var count1810 = 0; count1810 < repeat\_y; count1810++) {

for (var count1811 = 0; count1811 < repeat\_z; count1811++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.371, centerpoint\_y - sizemin \* 0.4995, centerpoint\_x - sizemid \* 0.335, centerpoint\_z + sizemax \* 0.42573, centerpoint\_y - sizemin \* 0.55, centerpoint\_x - sizemid \* 0.38];

box3();

if (make == true) {

bulbocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1812 = 0; count1812 < repeat\_x; count1812++) {

for (var count1813 = 0; count1813 < repeat\_y; count1813++) {

for (var count1814 = 0; count1814 < repeat\_z; count1814++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.4, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.49, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

box3();

if (make == true) {

hepatocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1815 = 0; count1815 < repeat\_x; count1815++) {

for (var count1816 = 0; count1816 < repeat\_y; count1816++) {

for (var count1817 = 0; count1817 < repeat\_z; count1817++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.32, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.34, centerpoint\_z + sizemax \* 0.39, centerpoint\_y - sizemin \* 0.45, centerpoint\_x - sizemid \* 0.6];

box3();

if (make == true) {

hepatocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1818 = 0; count1818 < repeat\_x; count1818++) {

for (var count1819 = 0; count1819 < repeat\_y; count1819++) {

for (var count1820 = 0; count1820 < repeat\_z; count1820++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['sweat'];

layer();

if (make == true) {

sudoricyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 5;

cedey = 6.187927353e+29 \* 5;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1821 = 0; count1821 < repeat\_x; count1821++) {

for (var count1822 = 0; count1822 < repeat\_y; count1822++) {

for (var count1823 = 0; count1823 < repeat\_z; count1823++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemid \* 0.299, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.4001, centerpoint\_x - sizemid \* 0.5];

box3();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 21) {

alpha\_cell();

} else if (randomizer < 32) {

delta\_cell();

} else if (randomizer < 37) {

gamma\_cell();

} else if (randomizer == 40) {

epsilon\_cell();

} else {

beta\_cell();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1824 = 0; count1824 < repeat\_x; count1824++) {

for (var count1825 = 0; count1825 < repeat\_y; count1825++) {

for (var count1826 = 0; count1826 < repeat\_z; count1826++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.51, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemid \* 0.299, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.4001, centerpoint\_x - sizemid \* 0.5];

box3();

if (make == true) {

acinicyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1827 = 0; count1827 < repeat\_x; count1827++) {

for (var count1828 = 0; count1828 < repeat\_y; count1828++) {

for (var count1829 = 0; count1829 < repeat\_z; count1829++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = (3.279601497e+28 \* 2.1 + 1.243773398e+30) + 1.243773398e+30;

thickabs2 = 3.279601497e+28 \* 2.1 + 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductA', 'ductB'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (var count1830 = 0; count1830 < repeat\_x; count1830++) {

for (var count1831 = 0; count1831 < repeat\_y; count1831++) {

for (var count1832 = 0; count1832 < repeat\_z; count1832++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.45];

box3();

if (make == true) {

thyrocyte();

} else {

box = [centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.55, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.6];

box3();

if (make == true) {

thyrocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.39, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.4];

box3();

if (make == true) {

parathyrocyte();

} else {

box = [centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.61];

box3();

if (make == true) {

parathyrocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.605, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemid \* 0.24];

box3();

if (make == true) {

adrenocyte\_1();

} else {

box = [centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.605, centerpoint\_x - sizemid \* 0.22, centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemid \* 0.24];

box3();

if (make == true) {

adrenocyte\_2();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.351, centerpoint\_y - sizemin \* 0.11, centerpoint\_x - sizemid \* 0.651, centerpoint\_z + sizemax \* 0.649, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.749];

box3();

if (make == true) {

adipocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.25, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.35, centerpoint\_y - sizemin \* 0.52, centerpoint\_x - sizemid \* 0.5];

box3();

if (make == true) {

lecyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.4, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.45];

box3();

if (make == true) {

thymocyte();

} else {

box = [centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.55, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.6];

box3();

if (make == true) {

thymocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 3;

cedey = 6.187927353e+29 \* 3;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

vertebrate\_skeleton();

if (make == true) {

hematocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.88, centerpoint\_y - sizemin \* 0.311, centerpoint\_x - sizemid \* 0.45, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.319, centerpoint\_x - sizemid \* 0.55];

box3();

if (make == true) {

salivacyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

smooth\_muscle\_cell();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

vertebrate\_connective\_all();

}

function vertebrate\_connective\_all() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

vertebrate\_connective\_1();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

if (make == false) {

make = false;

vertebrate\_connective\_1();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.39601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

vertebrate\_skeleton();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.39601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

vertebrate\_skeleton();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.922068735e+25 \* 1;

density\_parameter\_y = 1.922068735e+25 \* 1;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

bone();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

make = false;

vertebrate\_connective\_2();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

make = false;

vertebrate\_connective\_2();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1.1;

cedez = 6.187927353e+29 \* 1.1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 3.31, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 3.31, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 3.31, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 3.31, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 3.31, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 3.31, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 3.31, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 3.31, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 3.31];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 3.31];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 3.31];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 3.31];

vertebrate\_body();

if (make == true) {

randomizer = math\_random\_int(1, 10000);

if (randomizer < 2000) {

melanocyte();

} else if (randomizer < 4000) {

melanoblast();

} else {

epithelioblast();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2.77;

density\_parameter\_y = 1.033383868e+26 \* 2.3;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 6.187927353e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 6.187927353e+31, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z - 6.187927353e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 6.187927353e+31, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y - 6.187927353e+31, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 6.187927353e+31, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y + 6.187927353e+31, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 6.187927353e+31, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y, cursor1x + 6.187927353e+31];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 6.187927353e+31];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+31];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 6.187927353e+31];

vertebrate\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

if (par1 == true) {

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer < 51) {

elastin();

} else {

collagen();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 3;

density\_parameter\_y = 1.033383868e+26 \* 2.5;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_muscles();

if (make == true) {

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedez - 2);

repeat\_z = Math.round(sizemax / cedez - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.187927353e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

vertebrate\_body();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

vertebrate\_body();

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

}

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

vertebrate\_body();

}

if (make == true) {

dermatocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1.1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedez - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.187927353e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 7.5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 7.5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 7.5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 7.5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

vertebrate\_body();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

}

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

vertebrate\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

vertebrate\_body();

}

if (make == true) {

keratinocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 1.1;

density\_parameter\_y = 1.033383868e+26 \* 1.5;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

}

if (make == false) {

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.950341882e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.950341882e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

vertebrate\_body();

}

if (make == false) {

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.950341882e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.950341882e+30, cursor1x - 1.033383864e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

vertebrate\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.423223291e+26 \* 1;

density\_parameter\_y = 1.423223291e+26 \* 1;

density\_parameter\_z = 9.28189103e+25;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

exempt = [null];

if (make == true) {

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z + 3.403360044e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 3.403360044e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 3.403360044e+30, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 3.403360044e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 3.403360044e+30, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 3.403360044e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 3.403360044e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

vertebrate\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

vertebrate\_body();

if (make == false) {

if (make == false) {

loc1 = [cursor1z + 5.25973825e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 5.25973825e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 5.25973825e+30, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 5.25973825e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 5.25973825e+30, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 5.25973825e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 5.25973825e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

vertebrate\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

vertebrate\_body();

}

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

wax();

} else {

randomize\_lipid\_1();

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

unused\_variable = 5;

}

function vertebrate\_circulatory() {

bvar = 1;

box2 = [0, 0, 0, 1, 1, 1];

if (unused\_variable == 5) {

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.749, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.48, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.52]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.43, centerpoint\_z + sizemax \* 0.749, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.43, centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.47]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.51, centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['bronch', centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.43, centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemin \* 0.47]);

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemin \* 0.22, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.471];

loc1 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.43];

loc2 = [centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemin \* 0.47];

lung();

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.51, centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemin \* 0.55]);

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.329, centerpoint\_x - sizemin \* 0.509, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.78];

loc1 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.51];

loc2 = [centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemin \* 0.55];

lung();

respiratory();

dental\_1();

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.56, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.6]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.675, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.52, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.26, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.675, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.56, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.26, centerpoint\_x - sizemin \* 0.59]);

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6725, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.675, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6715, centerpoint\_y - sizemin \* 0.2417, centerpoint\_x - sizemin \* 0.5317, centerpoint\_z + sizemax \* 0.6725, centerpoint\_y - sizemin \* 0.2483, centerpoint\_x - sizemin \* 0.5383]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.2425, centerpoint\_x - sizemin \* 0.5325, centerpoint\_z + sizemax \* 0.6715, centerpoint\_y - sizemin \* 0.2475, centerpoint\_x - sizemin \* 0.5375]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.2446, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.2454, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.5346, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.5354]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.5325, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.5327]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6725, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.675, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6715, centerpoint\_y - sizemin \* 0.2417, centerpoint\_x - sizemin \* 0.5717, centerpoint\_z + sizemax \* 0.6725, centerpoint\_y - sizemin \* 0.2483, centerpoint\_x - sizemin \* 0.5783]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.2425, centerpoint\_x - sizemin \* 0.5725, centerpoint\_z + sizemax \* 0.6715, centerpoint\_y - sizemin \* 0.2475, centerpoint\_x - sizemin \* 0.5775]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.2446, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.2454, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.5746, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.5754]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.5725, centerpoint\_z + sizemax \* 0.6705, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.5727]);

}

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.6905, centerpoint\_y - sizemin \* 0.264, centerpoint\_x - sizemin \* 0.5315, centerpoint\_z + sizemax \* 0.6928, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.5385]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.6908, centerpoint\_y - sizemin \* 0.26, centerpoint\_x - sizemin \* 0.532, centerpoint\_z + sizemax \* 0.6925, centerpoint\_y - sizemin \* 0.264, centerpoint\_x - sizemin \* 0.538]);

}

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.6905, centerpoint\_y - sizemin \* 0.264, centerpoint\_x - sizemin \* 0.5715, centerpoint\_z + sizemax \* 0.6928, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemin \* 0.5785]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.6908, centerpoint\_y - sizemin \* 0.26, centerpoint\_x - sizemin \* 0.572, centerpoint\_z + sizemax \* 0.6925, centerpoint\_y - sizemin \* 0.264, centerpoint\_x - sizemin \* 0.578]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.567, centerpoint\_z + sizemax \* 0.6593, centerpoint\_y - sizemin \* 0.287, centerpoint\_x - sizemin \* 0.583]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.655, centerpoint\_y - sizemin \* 0.287, centerpoint\_x - sizemin \* 0.5705, centerpoint\_z + sizemax \* 0.6587, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.65515, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.571, centerpoint\_z + sizemax \* 0.6583, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.579]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.65665, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.6567, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.571, centerpoint\_z + sizemax \* 0.65665, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.5713]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6567, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.575, centerpoint\_z + sizemax \* 0.659, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.5753]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.567, centerpoint\_z + sizemax \* 0.6593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.583]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.583, centerpoint\_z + sizemax \* 0.6593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.781]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.542, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.297, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.667, centerpoint\_y - sizemin \* 0.287, centerpoint\_x - sizemin \* 0.541, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6666666666666666, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.54, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.287, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.297, centerpoint\_x - sizemin \* 0.546, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.2975, centerpoint\_x - sizemin \* 0.54615]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6685, centerpoint\_y - sizemin \* 0.297, centerpoint\_x - sizemin \* 0.54, centerpoint\_z + sizemax \* 0.6689, centerpoint\_y - sizemin \* 0.2975, centerpoint\_x - sizemin \* 0.546]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.297, centerpoint\_x - sizemin \* 0.54615, centerpoint\_z + sizemax \* 0.6674, centerpoint\_y - sizemin \* 0.2975, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.2975, centerpoint\_x - sizemin \* 0.542, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.55]);

}

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.7001, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.5, centerpoint\_z + sizemax \* 0.715, centerpoint\_y - sizemin \* 0.253, centerpoint\_x - sizemin \* 0.5199]);

tissues\_list.push(['SA node', centerpoint\_z + sizemax \* 0.7001, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5, centerpoint\_z + sizemax \* 0.725, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.5199]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.7001, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5199, centerpoint\_z + sizemax \* 0.705, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.7001, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.58, centerpoint\_z + sizemax \* 0.705, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.597]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.673, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.7001, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['AV node', centerpoint\_z + sizemax \* 0.671, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5515, centerpoint\_z + sizemax \* 0.673, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.558]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.671, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.61, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.625, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.63, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.635, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.65, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.66, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.555, centerpoint\_z + sizemax \* 0.61, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5599]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.61, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.555, centerpoint\_z + sizemax \* 0.615, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5599]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.555, centerpoint\_z + sizemax \* 0.625, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5599]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.63, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.555, centerpoint\_z + sizemax \* 0.64, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5599]);

}

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.317, centerpoint\_x - sizemin \* 0.6]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.6915, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.405]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.285, centerpoint\_x - sizemin \* 0.214, centerpoint\_z + sizemax \* 0.6915, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.405]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.214, centerpoint\_z + sizemax \* 0.6915, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.78, centerpoint\_x - sizemin \* 0.219, centerpoint\_z + sizemax \* 0.6915, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.255]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.25, centerpoint\_z + sizemax \* 0.6915, centerpoint\_y - sizemin \* 0.79, centerpoint\_x - sizemin \* 0.255]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.6, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.317, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.36, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.317, centerpoint\_x - sizemin \* 0.4]);

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.306, centerpoint\_x - sizemin \* 0.509, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.78];

loc1 = [centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.6];

loc2 = [centerpoint\_z + sizemax \* 0.693333333, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_x - sizemin \* 0.62];

vein();

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.306, centerpoint\_x - sizemin \* 0.22, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.471];

loc1 = [centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.307, centerpoint\_x - sizemin \* 0.36];

loc2 = [centerpoint\_z + sizemax \* 0.693333333, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_x - sizemin \* 0.4];

vein();

}

if (unused\_variable == 5) {

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.542, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.55, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.59]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.59, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.542]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.36, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.4]);

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.2979, centerpoint\_x - sizemin \* 0.509, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.78];

loc1 = [centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.59];

loc2 = [centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.62];

artery();

box = [centerpoint\_z + sizemax \* 0.59, centerpoint\_y - sizemin \* 0.2979, centerpoint\_x - sizemin \* 0.22, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.65, centerpoint\_x - sizemin \* 0.471];

loc1 = [centerpoint\_z + sizemax \* 0.6673, centerpoint\_y - sizemin \* 0.298, centerpoint\_x - sizemin \* 0.36];

loc2 = [centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.3069, centerpoint\_x - sizemin \* 0.4];

artery();

}

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.693333333, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.538]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.69, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.538]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.59666, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.22, centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.78]);

unused\_variable = 5;

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.59666, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.59666, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.79]);

unused\_variable = 5;

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.18, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.715, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.16, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.715, centerpoint\_x - sizemin \* 0.18]);

box = [centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.181];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.79]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.72, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.715, centerpoint\_x - sizemin \* 0.781]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.72, centerpoint\_z + sizemax \* 0.31333, centerpoint\_y - sizemin \* 0.715, centerpoint\_x - sizemin \* 0.76]);

box = [centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.719, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

if (unused\_variable == 5) {

gills();

}

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.715, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.313, centerpoint\_y - sizemin \* 0.77, centerpoint\_x - sizemin \* 0.215]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.25, centerpoint\_y - sizemin \* 0.765, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.31, centerpoint\_y - sizemin \* 0.77, centerpoint\_x - sizemin \* 0.215]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.23, centerpoint\_y - sizemin \* 0.765, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.25, centerpoint\_y - sizemin \* 0.77, centerpoint\_x - sizemin \* 0.215]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 0.251, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.7333, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.781]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.79]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.21, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.17, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.71, centerpoint\_x - sizemin \* 0.219]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.15, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.71, centerpoint\_x - sizemin \* 0.17]);

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.171];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.79]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.71, centerpoint\_x - sizemin \* 0.83]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.62, centerpoint\_y - sizemin \* 0.705, centerpoint\_x - sizemin \* 0.83, centerpoint\_z + sizemax \* 0.62333, centerpoint\_y - sizemin \* 0.71, centerpoint\_x - sizemin \* 0.85]);

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.8299, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.12604808e+30 \* 2.3;

cedey = 6.12604808e+30 \* 1;

cedez = 6.12604808e+30 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30];

make = true;

exempt = ['heart'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

thickabs = 1e+45;

thickabs2 = 6.12604808e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

tissues\_list.push(['heart', spawnerz, spawnery, spawnerx, spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30]);

}

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

make = false;

thickabs = 1e+45;

thickabs2 = 6.12604808e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

tissues\_list.push(['heart', spawnerz, spawnery, spawnerx, spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30]);

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 0.5 \* cedex;

spawnery = centerpoint\_y - 0.5 \* cedey;

spawnerz = centerpoint\_z + 0.5 \* cedez;

}

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = 1.23758547e+32;

thickabs2 = 6.187927353e+31;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = 1.23758547e+32;

thickabs2 = 6.187927353e+31;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

vertebrate\_circulatory2();

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 2.322329135e+26 \* 1;

cedey = 2.322329135e+26 \* 1;

cedez = 6.187927353e+28 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 5.569134618e+28, spawnery - 1.23758547e+26, spawnerx - 1.23758547e+26];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node', 'AV node', 'fiber 1'];

layer();

if (make == true) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

collagen();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 0.5 \* cedex;

spawnery = centerpoint\_y - 0.5 \* cedey;

spawnerz = centerpoint\_z + 0.5 \* cedez;

}

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

vascular();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 10;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

vertebrate\_body();

if (make == true) {

fibroblast();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

vertebrate\_muscles();

if (make == true) {

myoblast();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

cardioblast();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

cardioblast();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

}

function vertebrate\_circulatory2() {

bvar = 1;

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.75, centerpoint\_z + sizemax \* 0.657, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.755]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.75, centerpoint\_z + sizemax \* 0.657, centerpoint\_y - sizemin \* 0.79, centerpoint\_x - sizemin \* 0.755]);

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.28, centerpoint\_z + sizemax \* 0.73333333, centerpoint\_y - sizemin \* 0.291, centerpoint\_x - sizemin \* 0.538]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7333333, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.736, centerpoint\_y - sizemin \* 0.291, centerpoint\_x - sizemin \* 0.41]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.411];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.45, centerpoint\_z + sizemax \* 0.61, centerpoint\_y - sizemin \* 0.2837, centerpoint\_x - sizemin \* 0.458]);

box = [centerpoint\_z + sizemax \* 0.58, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemin \* 0.48, centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.629];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6549, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.583, centerpoint\_z + sizemax \* 0.6593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.781]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.65, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.6593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.791]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.64, centerpoint\_y - sizemin \* 0.29835, centerpoint\_x - sizemin \* 0.63, centerpoint\_z + sizemax \* 0.65593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.633]);

box = [centerpoint\_z + sizemax \* 0.58, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemin \* 0.48, centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.6331];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.64, centerpoint\_y - sizemin \* 0.29835, centerpoint\_x - sizemin \* 0.635, centerpoint\_z + sizemax \* 0.65593, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.638]);

box = [centerpoint\_z + sizemax \* 0.58, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemin \* 0.48, centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.64];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.64, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.65, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.791]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 0.651, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.792, centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.75, centerpoint\_x - sizemin \* 0.7921];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['lymph', centerpoint\_z + sizemax \* 0.7333333, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.411, centerpoint\_z + sizemax \* 0.736, centerpoint\_y - sizemin \* 0.291, centerpoint\_x - sizemin \* 0.414]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.202, centerpoint\_x - sizemin \* 0.41415, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.793, centerpoint\_x - sizemin \* 0.793];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

lymph();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6539, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.77, centerpoint\_z + sizemax \* 0.66, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.779]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.78];

box2 = [centerpoint\_z + sizemax \* 0.7, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemin \* 0.41415, centerpoint\_z + sizemax \* 0.749, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.44];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7333333, centerpoint\_y - sizemin \* 0.281, centerpoint\_x - sizemin \* 0.415, centerpoint\_z + sizemax \* 0.736, centerpoint\_y - sizemin \* 0.291, centerpoint\_x - sizemin \* 0.419]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemin \* 0.78, centerpoint\_z + sizemax \* 0.73, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.7991];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.6539, centerpoint\_y - sizemin \* 0.29535, centerpoint\_x - sizemin \* 0.781, centerpoint\_z + sizemax \* 0.66, centerpoint\_y - sizemin \* 0.30535, centerpoint\_x - sizemin \* 0.791]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

}

}

function respiratory() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['oral', 'visceral4', 'ear'];

layer();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

if (make == true) {

orocyte();

}

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['nasal'];

layer();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

if (make == true) {

mucocyte();

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['bronch', 'bronchiole', 'trachea'];

layer();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_body();

} else {

human\_body();

}

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 7) {

alveolocyte();

} else {

bronchocyte();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 2;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedey - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0.09;

target1 = ['bronch', 'bronchiole', 'trachea'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 1.856378206e+27;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.856378205e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = 2.7e+28;

thickabs2 = 6e+27;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['alveolus'];

layer();

if (make == true) {

elastin();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target1 = ['bronchiole', 'bronch'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.423223291e+30;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['oral', 'visceral4', 'ear'];

layer();

if (make == true) {

collagen();

} else {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 6.187927353e+27;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['alveolus'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

collagen();

} else {

elastin();

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target1 = ['bronchiole', 'bronch'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.423223291e+30;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['oral', 'visceral4', 'ear'];

layer();

if (make == true) {

collagen();

} else {

make = false;

thickabs = 3.093963676e+28;

thickabs2 = 6.187927353e+27;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['alveolus'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

collagen();

} else {

elastin();

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 1.856378206e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.856378205e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = 4.393428421e+28;

thickabs2 = 6.187927353e+27;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['bronchiole', 'bronch'];

layer();

if (make == true) {

elastin();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 4.08590675e+25, cursor1y - 1.54698183e+26, cursor1x - 4.08590675e+25];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = 3.15584295e+26;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['alveolus'];

layer();

if (make == true) {

randomize\_lipid\_1();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 3.093963676e+26 \* 2;

density\_parameter\_y = 3.093963676e+26 \* 2;

density\_parameter\_z = 1.856378206e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.856378205e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = 1.856378206e+27;

thickabs2 = 6.187927353e+26;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['alveolus'];

layer();

if (make == true) {

surfactant();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function dental\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 3;

used\_density\_parameter = 2.503635407e+25;

generation\_parameter\_x = Math.floor(sizemid / used\_density\_parameter);

generation\_parameter\_y = Math.floor(sizemin / used\_density\_parameter);

generation\_parameter\_z = Math.floor(sizemax / used\_density\_parameter);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - used\_density\_parameter;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - used\_density\_parameter;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

make = false;

if (bvar == 1) {

teeth\_1();

} else {

teeth\_2();

}

if (make == true) {

aluminium\_metallic\_();

}

list\_molecules.push(['aluminium(metallic)', cursor1z, cursor1y, cursor1x, cursor1z + used\_density\_parameter, cursor1y - used\_density\_parameter, cursor1x - used\_density\_parameter]);

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter / 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - used\_density\_parameter;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - used\_density\_parameter;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

make = false;

if (bvar == 1) {

teeth\_1();

} else {

teeth\_2();

}

if (make == true) {

aluminium\_metallic\_();

}

list\_molecules.push(['aluminium(metallic)', cursor1z, cursor1y, cursor1x, cursor1z + used\_density\_parameter, cursor1y - used\_density\_parameter, cursor1x - used\_density\_parameter]);

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1y = cursor1y + used\_density\_parameter / 2;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - used\_density\_parameter;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - used\_density\_parameter;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

make = false;

if (bvar == 1) {

teeth\_1();

} else {

teeth\_2();

}

if (make == true) {

aluminium\_metallic\_();

}

list\_molecules.push(['aluminium(metallic)', cursor1z, cursor1y, cursor1x, cursor1z + used\_density\_parameter, cursor1y - used\_density\_parameter, cursor1x - used\_density\_parameter]);

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

cursor1x = cursor1x + used\_density\_parameter / 2;

cursor1y = cursor1y + used\_density\_parameter;

cursor1z = cursor1z + used\_density\_parameter / 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - used\_density\_parameter;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - used\_density\_parameter;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + used\_density\_parameter;

set\_earth\_rotation();

make = false;

if (bvar == 1) {

teeth\_1();

} else {

teeth\_2();

}

if (make == true) {

aluminium\_metallic\_();

list\_molecules.push(['aluminium(metallic)', cursor1z, cursor1y, cursor1x, cursor1z + used\_density\_parameter, cursor1y - used\_density\_parameter, cursor1x - used\_density\_parameter]);

}

}

cursor1z = cursor1z - generation\_parameter\_z \* used\_density\_parameter;

}

cursor1y = cursor1y + generation\_parameter\_y \* used\_density\_parameter;

}

}

}

}

function vascular() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['atrium', 'ventricle', 'artery', 'vein'];

layer();

if (make == true) {

endotheliocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['lymph'];

layer();

if (make == true) {

lymphangiocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (bvar == 1) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.367531945e+30;

thickabs2 = 1.23758547e+29;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['capillary'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

ionocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 2;

cedez = 6.187927353e+29 \* 2;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1.5;

thickrel2 = 0.1;

target1 = ['artery'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == false) {

ionocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.3;

thickrel2 = 0.05;

target1 = ['artery'];

layer();

if (make == true) {

smooth\_muscle\_cell();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.03;

thickrel2 = 0.02;

target1 = ['vein'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

make = true;

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0.35;

target1 = ['artery'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0.04;

target1 = ['vein'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1;

thickrel2 = 0;

target1 = ['arteriole', 'venule'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.856378206e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['capillary', 'capillary 2', 'capillary 3', 'alveolar capillary'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

elastin();

} else {

collagen();

}

} else if (bvar == 1) {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1;

thickrel2 = 0.35;

target1 = ['artery'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.423223291e+30;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['artery', 'vein', 'atrium', 'ventricle', 'lymph'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0.35;

target1 = ['artery'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0.04;

target1 = ['vein'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1;

thickrel2 = 0;

target1 = ['arteriole', 'venule'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.856378206e+29;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['capillary', 'capillary 2', 'capillary 3', 'alveolar capillary'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

elastin();

} else {

collagen();

}

} else if (bvar == 1) {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1;

thickrel2 = 0.35;

target1 = ['artery'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

make = false;

thickabs = 1.423223291e+30;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['artery', 'vein', 'atrium', 'ventricle', 'lymph'];

layer();

if (make == true) {

collagen();

}

}

}

}

}

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.05;

thickrel2 = 0;

target1 = ['artery'];

layer();

if (make == true) {

elastin();

} else {

set\_earth\_rotation();

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.35;

thickrel2 = 0.3;

target1 = ['artery'];

layer();

if (make == true) {

elastin();

} else {

set\_earth\_rotation();

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.02;

thickrel2 = 0;

target1 = ['vein'];

layer();

if (make == true) {

elastin();

} else {

set\_earth\_rotation();

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.04;

thickrel2 = 0.03;

target1 = ['vein'];

layer();

if (make == true) {

elastin();

} else {

set\_earth\_rotation();

make = false;

thickabs = 6.187927353e+31;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

elastin();

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 4.950341882e+29 \* 1;

cedey = 2.475170941e+29 \* 1;

cedez = 4.950341882e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedey - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 4.950341881e+29, spawnery - 2.47517094e+29, spawnerx - 4.950341881e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1.5;

thickrel2 = 0.1;

target1 = ['artery', 'vein', 'arteriole', 'venule', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary'];

find2();

if (make == true) {

erythrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

}

function gills() {

bvar = 1;

if (unused\_variable == 5) {

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.03, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.25, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.79, centerpoint\_x - sizemin \* 0.255]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.03, centerpoint\_y - sizemin \* 0.785, centerpoint\_x - sizemin \* 0.75, centerpoint\_z + sizemax \* 0.3, centerpoint\_y - sizemin \* 0.79, centerpoint\_x - sizemin \* 0.755]);

spawnerx = centerpoint\_x - sizemin \* 0.25;

spawnery = centerpoint\_y - sizemin \* 0.79;

spawnerz = centerpoint\_z + sizemax \* 0.3;

for (countcg1 = 0; countcg1 < 25; countcg1++) {

spawnerz = centerpoint\_z - sizemax \* 0.01;

tissues\_list.push(['artery', spawnerz, spawnery, spawnerx, spawnerz + sizemax \* 0.001, spawnery - sizemin \* 0.03, spawnerx - sizemin \* 0.003]);

tissues\_list.push(['artery', spawnerz, spawnery - sizemin \* 0.03, spawnerx, spawnerz + sizemax \* 0.001, spawnery - sizemin \* 0.033, spawnerx - sizemin \* 0.5003]);

tissues\_list.push(['artery', spawnerz + (sizemax \* 0.001 + 6.187927353e+30), spawnery, spawnerx - sizemin \* 0.5, spawnerz + (sizemax \* 0.002 + 6.187927353e+30), spawnery - sizemin \* 0.03, spawnerx - sizemin \* 0.5003]);

tissues\_list.push(['artery', spawnerz + (sizemax \* 0.001 + 0), spawnery - sizemin \* 0.03, spawnerx - sizemin \* 0, spawnerz + (sizemax \* 0.001 + 6.187927353e+30), spawnery - sizemin \* 0.0303, spawnerx - sizemin \* 0.003]);

tissues\_list.push(['artery', spawnerz + (sizemax \* 0.001 + 6.187927353e+30), spawnery - sizemin \* 0.03, spawnerx - sizemin \* 0, spawnerz + (sizemax \* 0.002 + 6.187927353e+30), spawnery - sizemin \* 0.033, spawnerx - sizemin \* 0.5003]);

starter\_x = spawnerx;

starter\_y = spawnery - sizemin \* 0.033;

starter\_z = spawnerz + (sizemax \* 0.001 - 4.331549147e+30);

while (starter\_x > centerpoint\_x - sizemin \* 0.75) {

starter\_x = starter\_x - 2.475170941e+31;

tissues\_list.push(['vein', starter\_z, starter\_y, starter\_x, starter\_z + 4.331549147e+30, centerpoint\_y - sizemin \* 0.99, starter\_x - 4.331549147e+30]);

tissues\_list.push(['vein', starter\_z + 9.28189103e+30, starter\_y, starter\_x, starter\_z + 1.23758547e+31, centerpoint\_y - sizemin \* 0.99, starter\_x - 4.331549147e+30]);

tissues\_list.push(['vein', starter\_z + 9.28189103e+30, centerpoint\_y - (sizemin \* 0.99 - 6.187927353e+29), starter\_x, starter\_z + 1.23758547e+31, centerpoint\_y - sizemin \* 0.99, starter\_x - 4.331549147e+30]);

cursor1x = starter\_x - 4.331549147e+30;

cursor1y = starter\_y;

cursor1z = starter\_z;

while (cursor1y > centerpoint\_y - sizemin \* 0.989) {

cursor1y = cursor1y - 4.950341882e+30;

tissues\_list.push(['capillary', cursor1z, cursor1y, cursor1x, cursor1z + 6.187927353e+29, cursor1y - 6.187927353e+29, cursor1x - 1.23758547e+31]);

tissues\_list.push(['capillary', cursor1z + 9.28189103e+30, cursor1y, cursor1x, cursor1z + 9.900683765e+30, cursor1y - 6.187927353e+29, cursor1x - 1.23758547e+31]);

tissues\_list.push(['capillary', cursor1z + 9.900683765e+30, cursor1y, cursor1x - 1.23758547e+31, cursor1z + 1.05194765e+31, cursor1y - 6.187927353e+29, cursor1x - 1.299464744e+31]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 1.23758547e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 1.856378206e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 2.475170941e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 3.093963676e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 3.712756412e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 4.331549147e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 4.950341882e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 5.569134618e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 6.187927353e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 6.806720089e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 7.425512824e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 8.044305559e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 8.663098295e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 9.28189103e+30]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 9.900683765e+30, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 1.05194765e+31]);

tissues\_list.push(['capillary', cursor1z + 6.187927353e+29, cursor1y, cursor1x - 1.113826923e+31, cursor1z + 9.28189103e+30, cursor1y - 6.187927353e+29, cursor1x - 1.175706197e+31]);

}

}

}

}

}

function fluids\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

glucose();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

glucose();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

glucose();

}

}

} else {

glucose();

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

glucose();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

glucose();

}

}

} else {

glucose();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_fat();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_fat();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_fat();

}

}

} else {

randomize\_fat();

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_fat();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_fat();

}

}

} else {

randomize\_fat();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_amino\_acid();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_amino\_acid();

}

}

} else {

randomize\_amino\_acid();

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_amino\_acid();

}

}

} else {

randomize\_amino\_acid();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_ribonucleotide();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_ribonucleotide();

}

}

} else {

randomize\_ribonucleotide();

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_ribonucleotide();

}

}

} else {

randomize\_ribonucleotide();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_vitamin();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_vitamin();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_vitamin();

}

}

} else {

randomize\_vitamin();

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_vitamin();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_vitamin();

}

}

} else {

randomize\_vitamin();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

fluids\_2b();

}

function fluids\_2b() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function air\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 2.115113925e+26 \* 1;

density\_parameter\_y = 2.115113925e+26 \* 1;

density\_parameter\_z = 2.115113925e+26;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

make = false;

target1 = ['sweat', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'trachea', 'bronch', 'bronchiole', 'alveolus', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum', 'duct', 'ductA', 'ductB', 'ductC', 'duct1', 'bladder', 'ureter', 'nephridium', 'ear'];

find2();

if (make == true) {

randomize\_air\_molecule();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2.0001;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

layer();

if (make == false) {

thickabs = 1.25758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

}

} else {

if (make == true) {

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

thickabs = null;

thickabs2 = null;

thickrel = 2.0001;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == false) {

thickabs = 1.25758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function gland\_2() {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.896, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.352, centerpoint\_x - sizemid \* 0.502]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.89261, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.896, centerpoint\_y - sizemin \* 0.352, centerpoint\_x - sizemid \* 0.502]);

box = [centerpoint\_z + sizemax \* 0.8926, centerpoint\_y - sizemin \* 0.349, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.898579, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.515];

loc1 = [centerpoint\_z + sizemax \* 0.892, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.5];

loc1 = [centerpoint\_z + sizemax \* 0.896, centerpoint\_y - sizemin \* 0.352, centerpoint\_x - sizemid \* 0.502];

duct();

}

function human\_male() {

unused\_variable = 5;

bvar = 2;

box2 = [0, 0, 0, 1, 1, 1];

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 3;

sizemin = sizemid;

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_endh = string\_solar\_distance\_\_length\_parameter - 1;

for (var counth = 0; counth < repeat\_endh; counth++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.55]);

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

ductsize = 6.187927353e+30;

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.256]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.256]);

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.245, centerpoint\_z + sizemax \* 0.68, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.35];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.294, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.6]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.14, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.64]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.176, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.66, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.176, centerpoint\_x - sizemid \* 0.606]);

box = [centerpoint\_z + sizemax \* 0.571, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemid \* 0.57, centerpoint\_z + sizemax \* 0.669, centerpoint\_y - sizemin \* 0.174, centerpoint\_x - sizemid \* 0.75];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.727]);

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.289]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.239]);

}

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.228]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.158, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.54]);

}

if (unused\_variable == 5) {

tissues\_list.push(['duct1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.51034482758, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['bladder', centerpoint\_z + sizemax \* 0.51034482758, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.45, centerpoint\_z + sizemax \* 0.52413793103, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.55]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.522, centerpoint\_y - sizemin \* 0.344, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.52413793103, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.45]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.522, centerpoint\_y - sizemin \* 0.344, centerpoint\_x - sizemid \* 0.294, centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.244, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.4]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.244, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.28]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.285, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.32]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.325, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.35]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.4]);

spawnerx = centerpoint\_x - sizemid \* 0.244;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.28) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.285;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.32) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.325;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.35) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.355;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.4) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnerx > centerpoint\_x - sizemin \* 0.53) {

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.5, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnerx > centerpoint\_x - sizemin \* 0.54) {

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.5, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.499;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.54]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.499;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.54]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.534, centerpoint\_z + sizemax \* 0.509, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.539]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.51, centerpoint\_z + sizemax \* 0.509, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.534]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.53, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.534]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.53, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.534]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.529, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.55];

ductsize = 6.187927353e+29 \* 10;

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.524]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.524]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.515, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.5289];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5075, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.514]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5075, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.514]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.2795, centerpoint\_x - sizemid \* 0.5001, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.51499];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.22;

spawnery = centerpoint\_y - sizemin \* 0;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerx > centerpoint\_x - sizemin \* 0.78) {

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

tissues\_list.push(['sweat', spawnerz, centerpoint\_y, spawnerx, spawnerz + 6.187927353e+29 \* 3, centerpoint\_y - sizemin \* 0.004, spawnerx - 6.187927353e+29 \* 3]);

spawnerz = spawnerz + 6.187927353e+29 \* 50;

}

spawnerz = centerpoint\_z + sizemax \* 0.48;

spawnerx = spawnerx - 6.187927353e+29 \* 50;

}

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.8448275862, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.8448275862, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.85816091954, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.85816091954, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.86816091954, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.86816091954, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9025, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.49]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9025, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.51, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.48]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.9105, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.52]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.495]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.505, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.515]);

}

gland\_2();

musculature\_2();

human\_cells1();

human\_brain();

innervate\_3();

human\_circulatory();

human\_cells2();

fluids\_3();

air\_3();

}

function human\_female() {

unused\_variable = 5;

bvar = 3;

convert\_coordinates();

set\_organism\_rotation();

sizemid = sizemax / 3;

sizemin = sizemid;

box2 = [0, 0, 0, 1, 1, 1];

if (unused\_variable == 5) {

if (list\_\_universe\_string\_entities\_\_\_default.length >= 2) {

string\_solar\_\_distance\_parameter = 0;

string\_solar\_distance\_\_length\_parameter = list\_\_universe\_string\_entities\_\_\_default.length;

var repeat\_endh = string\_solar\_distance\_\_length\_parameter - 1;

for (var counth = 0; counth < repeat\_endh; counth++) {

if (string\_solar\_\_distance\_parameter - 1 < list\_\_universe\_string\_entities\_\_\_default.length) {

string\_solar\_distance\_\_\_selected\_ = list\_\_universe\_string\_entities\_\_\_default[(string\_solar\_\_distance\_parameter + 1) - 1];

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter + 1;

obtain\_string\_location();

if (string\_x < centerpoint\_x) {

if (string\_x > centerpoint\_x - sizemid) {

if (string\_y < centerpoint\_y) {

if (string\_y > centerpoint\_y - sizemin) {

if (string\_z < centerpoint\_z + sizemax) {

if (string\_z > centerpoint\_z) {

list.splice(string\_solar\_\_distance\_parameter - 1, 1);

string\_solar\_\_distance\_parameter = string\_solar\_\_distance\_parameter - 1;

}

}

}

}

}

}

}

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.55]);

tissues\_list.push(['visceral3', centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.5]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

ductsize = 6.187927353e+30;

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.256]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.256]);

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.245, centerpoint\_z + sizemax \* 0.68, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.35];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.294, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.6]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.14, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67796551724, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemid \* 0.64]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.676, centerpoint\_y - sizemin \* 0.176, centerpoint\_x - sizemid \* 0.606]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.66, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.67, centerpoint\_y - sizemin \* 0.176, centerpoint\_x - sizemid \* 0.606]);

box = [centerpoint\_z + sizemax \* 0.571, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemid \* 0.57, centerpoint\_z + sizemax \* 0.669, centerpoint\_y - sizemin \* 0.174, centerpoint\_x - sizemid \* 0.75];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.727]);

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.289]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.239]);

}

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.228]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.311]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.311, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.361]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.361, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.372]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.372, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.422]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.422, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.433]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.433, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.483]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.483, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.494]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.494, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.544]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.544, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.555]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.555, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.605]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.64896551724, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.605, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.616]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.616, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.666]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.239, centerpoint\_x - sizemid \* 0.666, centerpoint\_z + sizemax \* 0.60620689655, centerpoint\_y - sizemin \* 0.289, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.66896551724, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.677, centerpoint\_z + sizemax \* 0.58620689655, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.727]);

tissues\_list.push(['visceral2', centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.158, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.677]);

tissues\_list.push(['visceral1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.178, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.228, centerpoint\_x - sizemid \* 0.54]);

}

if (unused\_variable == 5) {

tissues\_list.push(['duct1', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.5, centerpoint\_z + sizemax \* 0.51034482758, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.51]);

tissues\_list.push(['bladder', centerpoint\_z + sizemax \* 0.51034482758, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.45, centerpoint\_z + sizemax \* 0.52413793103, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.55]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.522, centerpoint\_y - sizemin \* 0.344, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.52413793103, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.45]);

tissues\_list.push(['ureter', centerpoint\_z + sizemax \* 0.522, centerpoint\_y - sizemin \* 0.344, centerpoint\_x - sizemid \* 0.294, centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.3]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.54, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.244, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.4]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.244, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.28]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.285, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.32]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.325, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.35]);

tissues\_list.push(['nephridium', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.355, centerpoint\_z + sizemax \* 0.555, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.4]);

spawnerx = centerpoint\_x - sizemid \* 0.244;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.28) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.285;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.32) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.325;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.35) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

spawnerx = centerpoint\_x - sizemid \* 0.355;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.555;

while (spawnerx > sizemid \* 0.4) {

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

tissues\_list.push(['nephridium', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.58, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29]);

spawnery = spawnery - 6.187927353e+29 \* 6;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 6;

}

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnerx > centerpoint\_x - sizemin \* 0.53) {

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.5, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnerx > centerpoint\_x - sizemin \* 0.54) {

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductB', spawnerz, spawnery, spawnerx, centerpoint\_z + sizemax \* 0.5, spawnery - 6.187927353e+29 \* 10, spawnerx - 6.187927353e+29 \* 10]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerx = spawnerx - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.52;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.499;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.52]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.445;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.54]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

spawnerx = centerpoint\_x - sizemid \* 0.53;

spawnery = centerpoint\_y - (sizemin \* 0.3 + 6.187927353e+29 \* 10);

spawnerz = centerpoint\_z + sizemax \* 0.499;

while (spawnery > centerpoint\_y - sizemin \* 0.371) {

tissues\_list.push(['ductA', spawnerz, spawnery, spawnerx, spawnerz + 6.187927353e+29 \* 10, spawnery - 6.187927353e+29 \* 10, centerpoint\_x - sizemid \* 0.54]);

spawnery = spawnery - 6.187927353e+29 \* 20;

}

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.534, centerpoint\_z + sizemax \* 0.509, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.539]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.509, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.534]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.53, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.534]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.53, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.534]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.529, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.55];

ductsize = 6.187927353e+29 \* 10;

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.524]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.524]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.515, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.5289];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5075, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.514]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.5075, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.508, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.514]);

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.2795, centerpoint\_x - sizemid \* 0.5001, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.51499];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.505, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.505, centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.511]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.505, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.511, centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.67]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.505, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.665, centerpoint\_z + sizemax \* 0.507, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.67]);

tissues\_list.push(['ductC', centerpoint\_z + sizemax \* 0.505, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.665, centerpoint\_z + sizemax \* 0.566, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemid \* 0.67]);

}

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.22;

spawnery = centerpoint\_y - sizemin \* 0;

spawnerz = centerpoint\_z + sizemax \* 0.49;

while (spawnerx > centerpoint\_x - sizemin \* 0.78) {

while (spawnerz < centerpoint\_z + sizemax \* 0.8) {

tissues\_list.push(['sweat', spawnerz, centerpoint\_y, spawnerx, spawnerz + 6.187927353e+29 \* 3, centerpoint\_y - sizemin \* 0.004, spawnerx - 6.187927353e+29 \* 3]);

spawnerz = spawnerz + 6.187927353e+29 \* 50;

}

spawnerz = centerpoint\_z + sizemax \* 0.48;

spawnerx = spawnerx - 6.187927353e+29 \* 50;

}

}

if (unused\_variable == 5) {

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.68965517241, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.8448275862, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.8448275862, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.85816091954, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.85816091954, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.86816091954, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.86816091954, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9025, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.49]);

tissues\_list.push(['visceral4', centerpoint\_z + sizemax \* 0.9025, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemid \* 0.51, centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.2135, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.48]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.52, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.915, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemid \* 0.53]);

tissues\_list.push(['oral', centerpoint\_z + sizemax \* 0.91, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.9105, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.52]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.92, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.495]);

tissues\_list.push(['nasal', centerpoint\_z + sizemax \* 0.926, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemid \* 0.505, centerpoint\_z + sizemax \* 0.93, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.515]);

gland\_2();

}

if (unused\_variable == 5) {

tissues\_list.push(['vhomol', centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.53793103448, centerpoint\_y - sizemin \* 0.26, centerpoint\_x - sizemid \* 0.515]);

tissues\_list.push(['vhomol', centerpoint\_z + sizemax \* 0.53793103448, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemid \* 0.48, centerpoint\_z + sizemax \* 0.56551724137, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemid \* 0.52]);

tissues\_list.push(['uterus', centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemid \* 0.493, centerpoint\_z + sizemax \* 0.5535, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.507]);

tissues\_list.push(['uterus', centerpoint\_z + sizemax \* 0.535, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.46, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.54]);

tissues\_list.push(['isthmus', centerpoint\_z + sizemax \* 0.566, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.54, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.58]);

tissues\_list.push(['magnum', centerpoint\_z + sizemax \* 0.566, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.58, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.64]);

tissues\_list.push(['infundibulum', centerpoint\_z + sizemax \* 0.566, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.64, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.68]);

}

if (unused\_variable == 5) {

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.751, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.353]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.39, centerpoint\_x - sizemid \* 0.35, centerpoint\_z + sizemax \* 0.751, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.353]);

box = [centerpoint\_z + sizemax \* 0.73, centerpoint\_y - sizemin \* 0.385, centerpoint\_x - sizemid \* 0.3, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.4];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.65, centerpoint\_z + sizemax \* 0.751, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemid \* 0.653]);

tissues\_list.push(['duct', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.39, centerpoint\_x - sizemid \* 0.75, centerpoint\_z + sizemax \* 0.751, centerpoint\_y - sizemin \* 0.4, centerpoint\_x - sizemid \* 0.753]);

box = [centerpoint\_z + sizemax \* 0.73, centerpoint\_y - sizemin \* 0.385, centerpoint\_x - sizemid \* 0.6, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.5, centerpoint\_x - sizemid \* 0.7];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

duct();

}

unused\_variable = 5;

musculature\_2();

human\_cells1();

human\_brain();

innervate\_3();

human\_circulatory();

ovary\_2();

human\_cells2();

fluids\_3();

air\_3();

}

function musculature\_2() {

if (unused\_variable == 5) {

spawnerz = centerpoint\_z + (sizemax \* 0 - 0);

spawnery = centerpoint\_y - (sizemin \* 0 - 0);

spawnerx = centerpoint\_x - (sizemid \* 0 - 0);

muscle = false;

repeat\_endcg1 = Math.floor(sizemin / 2.475170941e+30);

for (countcg1 = 0; countcg1 < repeat\_endcg1; countcg1++) {

repeat\_endcg2 = Math.floor(sizemid / 2.475170941e+30);

for (countcg2 = 0; countcg2 < repeat\_endcg2; countcg2++) {

while (spawnerz < centerpoint\_z + (sizemax \* 1 - 0)) {

if (muscle == true) {

make = true;

exempt = [null];

loc1 = [spawnerz + 1.23758547e+30 \* 1, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 2, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_muscle();

if (make == true) {

myocyte();

} else {

muscle = false;

tendon\_2();

}

} else {

muscle = false;

tendon\_2();

}

} else {

make = true;

exempt = [null];

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 1, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_muscle();

if (make == true) {

loc1 = [spawnerz + 1.23758547e+30 \* 1, spawnery, spawnerx];

loc2 = [spawnerz + 1.23758547e+30 \* 2, spawnery - 1.23758547e+30 \* 1, spawnerx - 1.23758547e+30 \* 1];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_muscle();

if (make == true) {

tendon\_1();

muscle = true;

}

}

}

}

}

spawnerz = spawnerz + 1.23758547e+30;

}

tendon\_2();

spawnerz = centerpoint\_z + (sizemax \* 0.1 - 0);

spawnerx = spawnerx - 2.475170941e+30;

}

spawnery = spawnery - 2.475170941e+30;

spawnerx = spawnerx + 2.475170941e+30 \* Math.floor((sizemid / 10) / 2.475170941e+30);

}

}

}

function human\_cells1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['vhomol'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

vagocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['uterus'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

uterocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['isthmus'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

isthmocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['magnum'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

magnocyte();

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['infundibulum'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.56, centerpoint\_y - sizemin \* 0.339, centerpoint\_x - sizemid \* 0.675, centerpoint\_z + sizemax \* 0.571, centerpoint\_y - sizemin \* 0.351, centerpoint\_x - sizemid \* 0.69];

box3();

if (make == false) {

infundibulocyte();

}

}

} else {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductC'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

ductocyte();

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

endocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

layer();

if (make == true) {

nephrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'bladder'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

urocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

if (loc1[1] >= centerpoint\_y - sizemin \* 0.2) {

mammocyte();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductA'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

sustenocyte();

} else {

genocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductB'];

layer();

if (make == true) {

epycyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 3.279601497e+28 \* 2.5;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'sweat', 'ductA', 'ductB', 'ductC', 'nephridium', 'bladder', 'ureter', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.589, centerpoint\_x - sizemid \* 0.639, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.611, centerpoint\_x - sizemid \* 0.661];

box3();

if (make == false) {

collagen();

}

}

}

}

}

} else {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

if (loc1[1] > centerpoint\_y - sizemin \* 0.38) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 3.279601497e+28 \* 2.5;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct1', 'sweat', 'ductA', 'ductB', 'ductC', 'nephridium', 'bladder', 'ureter', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0.539, centerpoint\_y - sizemin \* 0.589, centerpoint\_x - sizemid \* 0.639, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.611, centerpoint\_x - sizemid \* 0.661];

box3();

if (make == false) {

collagen();

}

}

}

}

}

} else {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function ovary\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = 1;

thickbod2 = 0;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['infundibulum', null, null];

box = [centerpoint\_z + sizemax \* 0.566, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemid \* 0.68, centerpoint\_z + sizemax \* 0.57, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.7];

box3();

if (make == true) {

make = false;

layer();

if (make == true) {

ovocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

}

function human\_cells2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.4;

thickrel2 = 0;

target2 = null;

target1 = ['uterus'];

layer();

if (make == true) {

smooth\_muscle\_cell();

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['vhomol', 'isthmus', 'magnum', 'infundibulum'];

layer();

if (make == true) {

smooth\_muscle\_cell();

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0.02;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3', 'visceral4'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.529, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.55];

box3();

if (make == true) {

vesicocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemid \* 0.515, centerpoint\_z + sizemax \* 0.55, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.5289];

box3();

if (make == true) {

procyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.2795, centerpoint\_x - sizemid \* 0.5001, centerpoint\_z + sizemax \* 0.53, centerpoint\_y - sizemin \* 0.295, centerpoint\_x - sizemid \* 0.51499];

box3();

if (make == true) {

bulbocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.571, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemid \* 0.57, centerpoint\_z + sizemax \* 0.669, centerpoint\_y - sizemin \* 0.174, centerpoint\_x - sizemid \* 0.75];

box3();

if (make == true) {

hepatocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['sweat'];

layer();

if (make == true) {

sudoricyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 5;

cedey = 6.187927353e+29 \* 5;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.245, centerpoint\_z + sizemax \* 0.68, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.3];

box3();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 21) {

alpha\_cell();

} else if (randomizer < 32) {

delta\_cell();

} else if (randomizer < 37) {

gamma\_cell();

} else if (randomizer == 40) {

epsilon\_cell();

} else {

beta\_cell();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.6, centerpoint\_y - sizemin \* 0.1, centerpoint\_x - sizemid \* 0.245, centerpoint\_z + sizemax \* 0.68, centerpoint\_y - sizemin \* 0.17, centerpoint\_x - sizemid \* 0.3];

box3();

if (make == true) {

acinicyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = (3.279601497e+28 \* 2.1 + 1.243773398e+30) + 1.243773398e+30;

thickabs2 = 3.279601497e+28 \* 2.1 + 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['ductA', 'ductB'];

layer();

if (make == true) {

smooth\_muscle\_cell();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.86, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.53];

box3();

if (make == true) {

thyrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.85, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.4656, centerpoint\_z + sizemax \* 0.86, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.47];

box3();

if (make == true) {

parathyrocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.541, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.551, centerpoint\_y - sizemin \* 0.29, centerpoint\_x - sizemid \* 0.4];

box3();

if (make == true) {

adrenocyte\_1();

} else {

box = [centerpoint\_z + sizemax \* 0.541, centerpoint\_y - sizemin \* 0.27, centerpoint\_x - sizemid \* 0.25, centerpoint\_z + sizemax \* 0.551, centerpoint\_y - sizemin \* 0.28, centerpoint\_x - sizemid \* 0.4];

box3();

if (make == true) {

adrenocyte\_2();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.4445, centerpoint\_y - sizemin \* 0.2997, centerpoint\_x - sizemid \* 0.5197, centerpoint\_z + sizemax \* 0.5, centerpoint\_y - sizemin \* 0.5403, centerpoint\_x - sizemid \* 0.3713];

box3();

if (make == true) {

lecyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 3;

cedey = 6.187927353e+29 \* 3;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

human\_skeleton();

if (make == true) {

hematocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.8926, centerpoint\_y - sizemin \* 0.349, centerpoint\_x - sizemid \* 0.485, centerpoint\_z + sizemax \* 0.898579, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemid \* 0.515];

box3();

if (make == true) {

salivacyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['duct'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

smooth\_muscle\_cell();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['visceral1', 'visceral2', 'visceral3'];

box = [centerpoint\_z + sizemax \* 0.79, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemid \* 0.47, centerpoint\_z + sizemax \* 0.8, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemid \* 0.53];

box3();

if (make == true) {

thymocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.3805, centerpoint\_x - sizemid \* 0.299, centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.499, centerpoint\_x - sizemid \* 0.401];

box3();

if (make == true) {

make = false;

human\_body();

if (make == true) {

adipocyte();

}

} else {

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.3805, centerpoint\_x - sizemid \* 0.599, centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.499, centerpoint\_x - sizemid \* 0.701];

box3();

if (make == true) {

make = false;

human\_body();

if (make == true) {

adipocyte();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

human\_connective\_all();

}

function human\_connective\_all() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

if (make == false) {

make = false;

human\_connective\_1();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

layer();

if (make == true) {

collagen();

} else {

if (make == false) {

set\_earth\_rotation();

if (make == false) {

make = false;

human\_connective\_1();

if (make == true) {

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.39601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

human\_skeleton();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.39601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.1;

thickrel2 = 0;

target2 = null;

target1 = ['nasal', 'trachea'];

human\_skeleton();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.922068735e+25 \* 1;

density\_parameter\_y = 1.922068735e+25 \* 1;

density\_parameter\_z = 1.922068735e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

bone();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

make = false;

human\_connective\_2();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

make = false;

human\_connective\_2();

if (make == true) {

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1.1;

cedez = 6.187927353e+29 \* 1.1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedex - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

human\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 3.31, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 3.31, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 3.31, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 3.31, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 3.31, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 3.31, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 3.31, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 3.31, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 3.31];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 3.31];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 3.31];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 3.31];

human\_body();

if (make == true) {

randomizer = math\_random\_int(1, 10000);

if (randomizer < 2000) {

melanocyte();

} else if (randomizer < 4000) {

melanoblast();

} else {

epithelioblast();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2.77;

density\_parameter\_y = 1.033383868e+26 \* 2.3;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 6.187927353e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 6.187927353e+31, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z - 6.187927353e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 6.187927353e+31, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y - 6.187927353e+31, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 6.187927353e+31, cursor1x - 1.033383864e+26];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y + 6.187927353e+31, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 6.187927353e+31, cursor1x - 1.033383864e+26];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y, cursor1x + 6.187927353e+31];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 6.187927353e+31];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+31];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 6.187927353e+31];

human\_body();

if (make == false) {

par1 = true;

} else {

make = false;

}

if (par1 == true) {

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer < 51) {

elastin();

} else {

collagen();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 3;

density\_parameter\_y = 1.033383868e+26 \* 2.5;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_muscles();

if (make == true) {

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedez - 2);

repeat\_z = Math.round(sizemax / cedez - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 4.950341882e+29, spawnery - 2.475170941e+29, spawnerx - 4.950341882e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

human\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

human\_body();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

human\_body();

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

}

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz + 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery - 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx + 6.187927353e+29 \* 5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 5];

loc1 = [spawnerz - 6.187927353e+29 \* 5, spawnery + 6.187927353e+29 \* 5, spawnerx - 6.187927353e+29 \* 5];

human\_body();

}

if (make == true) {

dermatocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 1.1;

cedey = 6.187927353e+29 \* 1.1;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 2);

repeat\_y = Math.round(sizemin / cedex - 2);

repeat\_z = Math.round(sizemax / cedez - 2);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.187927353e+29, spawnerx - 6.18792735e+29];

make = true;

exempt = [null];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

human\_body();

if (make == false) {

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery, spawnerx];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery - 6.187927353e+29 \* 7.5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery + 6.187927353e+29 \* 7.5, spawnerx];

loc2 = [spawnerz + 6.187927353e+29, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx + 6.187927353e+29 \* 7.5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

human\_body();

loc1 = [spawnerz, spawnery, spawnerx - 6.187927353e+29 \* 7.5];

loc2 = [spawnerz + 6.187927353e+29, spawnery - 6.187927353e+29, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

human\_body();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

human\_body();

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

}

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz + 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery - 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) + 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx + 6.187927353e+29 \* 7.5];

human\_body();

loc2 = [(spawnerz + 6.187927353e+29) - 6.187927353e+29 \* 7.5, (spawnery - 6.187927353e+29) + 6.187927353e+29 \* 7.5, (spawnerx - 6.187927353e+29) - 6.187927353e+29 \* 7.5];

loc1 = [spawnerz - 6.187927353e+29 \* 7.5, spawnery + 6.187927353e+29 \* 7.5, spawnerx - 6.187927353e+29 \* 7.5];

human\_body();

}

if (make == true) {

keratinocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 1.1;

density\_parameter\_y = 1.033383868e+26 \* 1.5;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

}

if (make == false) {

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.950341882e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.950341882e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = true;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

human\_body();

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 4.640945515e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.640945515e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.640945515e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x - 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) - 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y + 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) + 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z - 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

loc1 = [cursor1z + 4.640945515e+30, cursor1y - 4.640945515e+30, cursor1x + 4.640945515e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.640945515e+30, (cursor1y - 1.033383864e+26) - 4.640945515e+30, (cursor1x - 1.033383864e+26) + 4.640945515e+30];

human\_body();

}

if (make == false) {

if (make == false) {

par1 = false;

make = false;

exempt = [null];

loc1 = [cursor1z + 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) - 4.950341882e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 4.950341882e+30, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, (cursor1y - 1.033383864e+26) + 4.950341882e+30, cursor1x - 1.033383864e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 4.950341882e+30];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x - 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) - 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y + 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) + 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z - 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) - 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

loc1 = [cursor1z + 4.950341882e+30, cursor1y - 4.950341882e+30, cursor1x + 4.950341882e+30];

loc2 = [(cursor1z + 3.279601495e+28) + 4.950341882e+30, (cursor1y - 1.033383864e+26) - 4.950341882e+30, (cursor1x - 1.033383864e+26) + 4.950341882e+30];

human\_body();

}

if (make == true) {

set\_earth\_rotation();

collagen();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.423223291e+26 \* 1;

density\_parameter\_y = 1.423223291e+26 \* 1;

density\_parameter\_z = 9.28189103e+25;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

exempt = [null];

if (make == true) {

human\_body();

if (make == false) {

loc1 = [cursor1z + 3.403360044e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 3.403360044e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 3.403360044e+30, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 3.403360044e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 3.403360044e+30, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 3.403360044e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 3.403360044e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

human\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

human\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

human\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

human\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x - 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) - 3.403360044e+30];

human\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y + 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) + 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

human\_body();

loc1 = [cursor1z - 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) - 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

human\_body();

loc1 = [cursor1z + 3.403360044e+30, cursor1y - 3.403360044e+30, cursor1x + 3.403360044e+30];

loc2 = [(cursor1z + 1.56e+26) + 3.403360044e+30, (cursor1y - 1.84698183e+26) - 3.403360044e+30, (cursor1x - 1.82322329e+26) + 3.403360044e+30];

human\_body();

if (make == false) {

if (make == false) {

loc1 = [cursor1z + 5.25973825e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 5.25973825e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 5.25973825e+30, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 5.25973825e+30, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 5.25973825e+30, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 5.25973825e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 5.25973825e+30];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

human\_body();

if (make == false) {

loc1 = [cursor1z - 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

human\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

human\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

human\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

human\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x - 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) - 5.25973825e+30];

human\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y + 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) + 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

human\_body();

loc1 = [cursor1z - 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) - 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

human\_body();

loc1 = [cursor1z + 5.25973825e+30, cursor1y - 5.25973825e+30, cursor1x + 5.25973825e+30];

loc2 = [(cursor1z + 1.56e+26) + 5.25973825e+30, (cursor1y - 1.84698183e+26) - 5.25973825e+30, (cursor1x - 1.82322329e+26) + 5.25973825e+30];

human\_body();

}

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

wax();

} else {

randomize\_lipid\_1();

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function human\_circulatory2() {

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.74, centerpoint\_z + sizemax \* 0.761, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.75]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.74, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.75]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

}

function human\_circulatory() {

box2 = [0, 0, 0, 1, 1, 1];

if (unused\_variable == 5) {

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.8925, centerpoint\_y - sizemin \* 0.205, centerpoint\_x - sizemin \* 0.485, centerpoint\_z + sizemax \* 0.898, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.52]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.782, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.46, centerpoint\_z + sizemax \* 0.898, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.782, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.806, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.46]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.782, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.3, centerpoint\_z + sizemax \* 0.806, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.4]);

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemin \* 0.245, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.45];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

lung();

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.782, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.54, centerpoint\_z + sizemax \* 0.806, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.65]);

tissues\_list.push(['trachea', centerpoint\_z + sizemax \* 0.782, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.65, centerpoint\_z + sizemax \* 0.806, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.7]);

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.76];

box2 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.16, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.65];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

lung();

box2 = [0, 0, 0, 1, 1, 1];

}

respiratory();

dental\_1();

unused\_variable = 5;

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.73448275862, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.73448275862, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.56, centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.56, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.62]);

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.766, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemin \* 0.52, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.303, centerpoint\_x - sizemin \* 0.523, centerpoint\_z + sizemax \* 0.766, centerpoint\_y - sizemin \* 0.317, centerpoint\_x - sizemin \* 0.537]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.305, centerpoint\_x - sizemin \* 0.525, centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.315, centerpoint\_x - sizemin \* 0.535]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.52, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.3101, centerpoint\_x - sizemin \* 0.54]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.5301]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.3101, centerpoint\_x - sizemin \* 0.525, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.5251]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.766, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.59]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.303, centerpoint\_x - sizemin \* 0.573, centerpoint\_z + sizemax \* 0.766, centerpoint\_y - sizemin \* 0.317, centerpoint\_x - sizemin \* 0.587]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.305, centerpoint\_x - sizemin \* 0.575, centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.315, centerpoint\_x - sizemin \* 0.585]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.57, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.3101, centerpoint\_x - sizemin \* 0.59]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemin \* 0.58, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.31, centerpoint\_x - sizemin \* 0.5801]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7635, centerpoint\_y - sizemin \* 0.3101, centerpoint\_x - sizemin \* 0.575, centerpoint\_z + sizemax \* 0.7637, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.5751]);

}

if (unused\_variable == 5) {

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.776, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.61, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.7745, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.605, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.6, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.62]);

}

if (unused\_variable == 5) {

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.7745, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.335, centerpoint\_x - sizemin \* 0.505]);

tissues\_list.push(['atrium', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.335, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.354, centerpoint\_x - sizemin \* 0.51]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.34, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.515]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.52, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.25, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.245, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.535, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.24, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2299, centerpoint\_x - sizemin \* 0.5425, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5426]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.75725, centerpoint\_y - sizemin \* 0.2299, centerpoint\_x - sizemin \* 0.535, centerpoint\_z + sizemax \* 0.7573, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5425]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2299, centerpoint\_x - sizemin \* 0.5425, centerpoint\_z + sizemax \* 0.7546, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5426]);

}

if (unused\_variable == 5) {

tissues\_list.push(['ventricle', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.58, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.335, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7515, centerpoint\_y - sizemin \* 0.335, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.335, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7565, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.7566, centerpoint\_y - sizemin \* 0.3455, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.5975, centerpoint\_z + sizemax \* 0.7565, centerpoint\_y - sizemin \* 0.3455, centerpoint\_x - sizemin \* 0.5976]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7566, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.3455, centerpoint\_x - sizemin \* 0.586]);

}

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.58, centerpoint\_z + sizemax \* 0.7805, centerpoint\_y - sizemin \* 0.325, centerpoint\_x - sizemin \* 0.595]);

tissues\_list.push(['SA node', centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.53, centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.33, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.55, centerpoint\_z + sizemax \* 0.7805, centerpoint\_y - sizemin \* 0.325, centerpoint\_x - sizemin \* 0.58]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.769, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.325, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['AV node', centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.32, centerpoint\_x - sizemin \* 0.5515, centerpoint\_z + sizemax \* 0.769, centerpoint\_y - sizemin \* 0.325, centerpoint\_x - sizemin \* 0.558]);

tissues\_list.push(['fiber 1', centerpoint\_z + sizemax \* 0.7345, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.765, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.555]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.755, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.553]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.745, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.553]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.743, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.553]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.5501, centerpoint\_z + sizemax \* 0.739, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.553]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.755, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5595]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.754, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5595]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.745, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.749, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5595]);

tissues\_list.push(['fiber 2', centerpoint\_z + sizemax \* 0.735, centerpoint\_y - sizemin \* 0.23, centerpoint\_x - sizemin \* 0.553, centerpoint\_z + sizemax \* 0.74, centerpoint\_y - sizemin \* 0.235, centerpoint\_x - sizemin \* 0.5595]);

}

if (unused\_variable == 5) {

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.18, centerpoint\_x - sizemin \* 0.6, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.18, centerpoint\_x - sizemin \* 0.475, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.6]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.18, centerpoint\_x - sizemin \* 0.455, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.475]);

box = [centerpoint\_z + sizemax \* 0.7199, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemin \* 0.244, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.4751];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.1799, centerpoint\_x - sizemin \* 0.6, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.18, centerpoint\_x - sizemin \* 0.62]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.773, centerpoint\_y - sizemin \* 0.165, centerpoint\_x - sizemin \* 0.6, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.1799, centerpoint\_x - sizemin \* 0.62]);

box = [centerpoint\_z + sizemax \* 0.7199, centerpoint\_y - sizemin \* 0.089, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.8071, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.761];

box2 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.18, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.65];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

}

if (unused\_variable == 5) {

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.535, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.2299, centerpoint\_x - sizemin \* 0.55]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.5, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.535]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.485, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.5]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.4752, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.485]);

box = [centerpoint\_z + sizemax \* 0.7198, centerpoint\_y - sizemin \* 0.09, centerpoint\_x - sizemin \* 0.243, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.4851];

box2 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.221, centerpoint\_x - sizemin \* 0.4751, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.65];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.55, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.22, centerpoint\_x - sizemin \* 0.57]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.199, centerpoint\_x - sizemin \* 0.55, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.57]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.7545, centerpoint\_y - sizemin \* 0.181, centerpoint\_x - sizemin \* 0.55, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.199, centerpoint\_x - sizemin \* 0.57]);

box = [centerpoint\_z + sizemax \* 0.7198, centerpoint\_y - sizemin \* 0.0889, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.807156, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.761];

box2 = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.545, centerpoint\_z + sizemax \* 0.807, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.65];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.12604808e+30 \* 2.3;

cedey = 6.12604808e+30 \* 1;

cedez = 6.12604808e+30 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30];

make = true;

exempt = ['heart'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

thickabs = 1e+45;

thickabs2 = 6.12604808e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

tissues\_list.push(['heart', spawnerz, spawnery, spawnerx, spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30]);

}

}

} else {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

make = false;

thickabs = 1e+45;

thickabs2 = 6.12604808e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

tissues\_list.push(['heart', spawnerz, spawnery, spawnerx, spawnerz + 6.12604808e+30, spawnery - 6.12604808e+30, spawnerx - 6.12604808e+30]);

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = 1.23758547e+32;

thickabs2 = 6.187927353e+31;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 0.5 \* cedex;

spawnery = centerpoint\_y - 0.5 \* cedey;

spawnerz = centerpoint\_z + 0.5 \* cedez;

}

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = 1.23758547e+32;

thickabs2 = 6.187927353e+31;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['atrium', 'ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.3455, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemin \* 0.585, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.61]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.753, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.61, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.75]);

unused\_variable = 5;

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.345, centerpoint\_x - sizemin \* 0.49, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.515]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.35, centerpoint\_x - sizemin \* 0.25, centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.49]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.48, centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.49]);

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.165, centerpoint\_x - sizemin \* 0.46, centerpoint\_z + sizemax \* 0.78175, centerpoint\_y - sizemin \* 0.3701, centerpoint\_x - sizemin \* 0.65];

box2 = [0, 0, 0, 1, 1, 1];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.36, centerpoint\_x - sizemin \* 0.651, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.661]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.651, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.661]);

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.165, centerpoint\_x - sizemin \* 0.46, centerpoint\_z + sizemax \* 0.78175, centerpoint\_y - sizemin \* 0.3701, centerpoint\_x - sizemin \* 0.662];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.663, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.673]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.663, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.673]);

box = [centerpoint\_z + sizemax \* 0.72, centerpoint\_y - sizemin \* 0.165, centerpoint\_x - sizemin \* 0.46, centerpoint\_z + sizemax \* 0.78175, centerpoint\_y - sizemin \* 0.3701, centerpoint\_x - sizemin \* 0.674];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.435, centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.45]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.435, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.45]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.219, centerpoint\_z + sizemax \* 0.7761, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.781];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['lymph', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.25, centerpoint\_z + sizemax \* 0.7715, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.26]);

tissues\_list.push(['lymph', centerpoint\_z + sizemax \* 0.75, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.25, centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.26]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [centerpoint\_z + sizemax \* 0.725, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.27, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 0.371, centerpoint\_x - sizemin \* 0.76];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

lymph();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.3, centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.31]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.3, centerpoint\_z + sizemax \* 0.79, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.31]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.6];

box2 = [centerpoint\_z + sizemax \* 0.725, centerpoint\_y - sizemin \* 0.2, centerpoint\_x - sizemin \* 0.4, centerpoint\_z + sizemax \* 0.8, centerpoint\_y - sizemin \* 0.373, centerpoint\_x - sizemin \* 0.46];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

vein();

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.7, centerpoint\_z + sizemax \* 0.761, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.71]);

tissues\_list.push(['artery', centerpoint\_z + sizemax \* 0.76, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.7, centerpoint\_z + sizemax \* 0.77, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.71]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 0.711];

loc1 = [(tissues\_list.slice(-1)[0])[1], (tissues\_list.slice(-1)[0])[2], (tissues\_list.slice(-1)[0])[3]];

loc2 = [(tissues\_list.slice(-1)[0])[4], (tissues\_list.slice(-1)[0])[5], (tissues\_list.slice(-1)[0])[6]];

artery();

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.78, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.401, centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.41]);

tissues\_list.push(['vein', centerpoint\_z + sizemax \* 0.781, centerpoint\_y - sizemin \* 0.361, centerpoint\_x - sizemin \* 0.401, centerpoint\_z + sizemax \* 0.79, centerpoint\_y - sizemin \* 0.37, centerpoint\_x - sizemin \* 0.41]);

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0, centerpoint\_z + sizemax \* 1, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box2 = [centerpoint\_z + sizemax \* 0.725, centerpoint\_y - sizemin \* 0.3, centerpoint\_x - sizemin \* 0.7, centerpoint\_z + sizemax \* 0.8, centerpoint\_y - sizemin \* 0.373, centerpoint\_x - sizemin \* 0.751];

}

human\_circulatory2();

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 2.322329135e+26 \* 1;

cedey = 2.322329135e+26 \* 1;

cedez = 6.187927353e+28 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 5.569134618e+28, spawnery - 1.23758547e+26, spawnerx - 1.23758547e+26];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node', 'AV node', 'fiber 1'];

layer();

if (make == true) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

collagen();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

cedex = 1.2375855e+30 \* 1;

cedey = 1.2375855e+30 \* 1;

cedez = 1.237585532e+31 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - 0.5 \* cedex;

spawnery = centerpoint\_y - 0.5 \* cedey;

spawnerz = centerpoint\_z + 0.5 \* cedez;

}

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927353e+30, spawnery - 1.23758547e+30, spawnerx - 1.23758547e+30];

make = true;

exempt = ['heart', 'SA node', 'AV node', 'fiber 1', 'fiber 2'];

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['SA node'];

find2();

if (make == true) {

cardiomyocyte\_1();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['AV node'];

find2();

if (make == true) {

cardiomyocyte\_2();

} else {

if (make == false) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['fiber 1', 'fiber 2'];

find2();

if (make == true) {

cardiomyocyte\_3();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

make = false;

box = [centerpoint\_z + sizemax \* 0, centerpoint\_y - sizemin \* 0, centerpoint\_x - sizemin \* 0.5505, centerpoint\_z + sizemax \* 0.5595, centerpoint\_y - sizemin \* 1, centerpoint\_x - sizemin \* 1];

box3();

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 1e+30;

thickrel2 = 0.05;

target2 = null;

target1 = ['artery', 'vein'];

layer();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer < 31) {

cardiomyocyte\_3();

} else {

cardiomyocyte\_4();

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

vascular();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 10;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

human\_body();

if (make == true) {

fibroblast();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['nephridium'];

human\_muscles();

if (make == true) {

myoblast();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29 \* 7;

cedey = 6.187927353e+29 \* 7;

cedez = 6.187927353e+29 \* 1;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.5;

thickrel2 = 0;

target2 = null;

target1 = ['ventricle'];

layer();

if (make == true) {

cardioblast();

} else {

if (make == false) {

make = false;

thickabs = null;

thickabs2 = null;

thickbod = null;

thickbod2 = null;

thickrel = 0.25;

thickrel2 = 0;

target2 = null;

target1 = ['atrium'];

layer();

if (make == true) {

cardioblast();

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedex;

}

}

unused\_variable = 5;

}

function fluids\_3() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

glucose();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

glucose();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

glucose();

}

}

} else {

glucose();

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

glucose();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

glucose();

}

}

} else {

glucose();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_fat();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_fat();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_fat();

}

}

} else {

randomize\_fat();

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_fat();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_fat();

}

}

} else {

randomize\_fat();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_amino\_acid();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_amino\_acid();

}

}

} else {

randomize\_amino\_acid();

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_amino\_acid();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_amino\_acid();

}

}

} else {

randomize\_amino\_acid();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_ribonucleotide();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_ribonucleotide();

}

}

} else {

randomize\_ribonucleotide();

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_ribonucleotide();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_ribonucleotide();

}

}

} else {

randomize\_ribonucleotide();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

randomize\_vitamin();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

randomize\_vitamin();

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_vitamin();

}

}

} else {

randomize\_vitamin();

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

randomize\_vitamin();

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

randomize\_vitamin();

}

}

} else {

randomize\_vitamin();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (make == true) {

make = false;

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

} else {

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NADH();

} else if (randomizer <= 40) {

FADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

fluids\_3b();

}

function fluids\_3b() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.922068734e+25, cursor1y - 1.922068734e+25, cursor1x - 1.922068734e+25];

make = false;

target1 = ['eye1', 'eye2', 'atrium', 'ventricle', 'heart', 'cap', 'SA node', 'AV node', 'fiber 1', 'fiber 2', 'capillary', 'capillary 2', 'capillary 3', 'alveolar capillary', 'lymph', 'cochlea', 'arteriole', 'venule'];

find2();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

vertebrate\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

thickabs = 1.23758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (make == true) {

make = false;

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

vertebrate\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 1.47433486e+25, cursor1y - 1.47433486e+25, cursor1x - 1.47433486e+25];

thickabs = null;

thickabs2 = null;

thickrel = 1e+22;

thickrel2 = 2;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

} else {

thickabs = 1e+42;

thickabs2 = 1.23758547e+33;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == true) {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

} else {

if (((cursor1x > brainx - 3.527118591e+32 && cursor1x < brainx) && (cursor1y > brainy - 2.475170941e+32 && cursor1y < brainy)) && (cursor1z < brainz + 3.520930664e+32 && cursor1z > brainz)) {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

} else {

randomize\_ECF();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function air\_3() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 2.115113925e+26 \* 1;

density\_parameter\_y = 2.115113925e+26 \* 1;

density\_parameter\_z = 2.115113925e+26;

cursor1x = cursor1x - density\_parameter\_x / 2;

cursor1y = cursor1y - density\_parameter\_y / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

make = false;

target1 = ['sweat', 'visceral1', 'visceral2', 'visceral3', 'visceral4', 'oral', 'nasal', 'trachea', 'bronch', 'bronchiole', 'alveolus', 'vhomol', 'uterus', 'isthmus', 'magnum', 'infundibulum', 'duct', 'ductA', 'ductB', 'ductC', 'duct1', 'bladder', 'ureter', 'nephridium', 'ear'];

find2();

if (make == true) {

randomize\_air\_molecule();

} else {

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (cursor1z > centerpoint\_z + sizemax \* 0.75) {

if (make == true) {

human\_body();

if (make == false) {

loc1 = [cursor1z + 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 6.806720089e+31, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 6.806720089e+31, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 6.806720089e+31];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

if (make == false) {

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x - 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) - 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y + 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) + 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z - 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) - 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

loc1 = [cursor1z + 6.806720089e+31, cursor1y - 6.806720089e+31, cursor1x + 6.806720089e+31];

loc2 = [(cursor1z + 1.56e+26) + 6.806720089e+31, (cursor1y - 1.84698183e+26) - 6.806720089e+31, (cursor1x - 1.82322329e+26) + 6.806720089e+31];

human\_body();

if (make == false) {

thickabs = null;

thickabs2 = null;

thickrel = 2.0001;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

layer();

if (make == false) {

thickabs = 1.25758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

}

} else {

if (make == true) {

human\_body();

if (make == false) {

loc1 = [cursor1z + 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y, cursor1x];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, cursor1y - 1.84698183e+26, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y - 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) - 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y + 1.23758547e+33, cursor1x];

loc2 = [cursor1z + 1.56e+26, (cursor1y - 1.84698183e+26) + 1.23758547e+33, cursor1x - 1.82322329e+26];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x + 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z, cursor1y, cursor1x - 1.23758547e+33];

loc2 = [cursor1z + 1.56e+26, cursor1y - 1.84698183e+26, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x - 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) - 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y + 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) + 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z - 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) - 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

loc1 = [cursor1z + 1.23758547e+33, cursor1y - 1.23758547e+33, cursor1x + 1.23758547e+33];

loc2 = [(cursor1z + 1.56e+26) + 1.23758547e+33, (cursor1y - 1.84698183e+26) - 1.23758547e+33, (cursor1x - 1.82322329e+26) + 1.23758547e+33];

human\_body();

if (make == false) {

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 2.115113923e+26, cursor1y - 2.115113923e+26, cursor1x - 2.115113923e+26];

thickabs = null;

thickabs2 = null;

thickrel = 2.0001;

thickrel2 = 0;

thickbod = null;

thickbod2 = null;

target1 = ['artery'];

layer();

if (make == false) {

thickabs = 1.25758547e+33;

thickabs2 = 0;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

target1 = ['vein', 'capillary'];

layer();

if (make == false) {

randomize\_air\_molecule();

}

}

}

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function fly\_brain\_in() {

brain1 = true;

}

function mouse\_brain\_in() {

brain1 = true;

}

function obtain\_fly\_neuron() {

neuron1 = "brains";

}

function obtain\_mouse\_neuron() {

neuron1 = "brains";

}

function randomize\_synaptic\_protein() {

randomizer = math\_random\_int(1, 140);

if (randomizer <= 10) {

randomizer = math\_random\_int(1, 70);

if (randomizer <= 10) {

peptidesequence = '1 mvrfgdelgg ryggpggger argggaggag gpgpgglqpg qrvlykqsia qrartmalyn 61 pipvkqncft vnrslfvfse dnvvrkyakr itewppfeym ilatiianci vlaleqhlpd 121 gdktpmserl ddtepyfigi fcfeagikii algfvfhkgs ylrngwnvmd fvvvltgila 181 tagtdfdlrt lravrvlrpl klvsgipslq vvlksimkam vpllqiglll ffailmfaii 241 glefymgkfh kacfpnstda epvgdfpcgk eaparlcegd tecreywpgp nfgitnfdni 301 lfailtvfqc itmegwtdil yntndaagnt wnwlyfipli iigsffmlnl vlgvlsgefa 361 kerervenrr aflklrrqqq ierelngyle wifkaeevml aeedrnaeek spldvlkraa 421 tkksrndlih aeegedrfad lcavgspfar aslksgktes ssyfrrkekm frffirrmvk 481 aqsfywvvlc vvalntlcva mvhynqprrl tttlyfaefv flglfltems lkmyglgprs 541 yfrssfncfd fgvivgsvfe vvwaaikpgs sfgisvlral rllrifkvtk ywsslrnlvv 601 sllnsmksii sllfllflfi vvfallgmql fggqfnfqde tpttnfdtfp aailtvfqil 661 tgedwnavmy hgiesqggvs kgmfssfyfi vltlfgnytl lnvflaiavd nlanaqeltk 721 deeemeeaan qklalqkake vaevspmsaa nisiaarqqn sakarsvweq rasqlrlqnl 781 rascealyse mdpeerlrfa ttrhlrpdmk thldrplvve lgrdgargpv ggkarpeaae 841 apegvdpprr hhrhrdkdkt paagdqdrae apkaesgepg areerprphr shskeaagpp 901 earsergrgp gpeggrrhhr rgspeeaaer eprrhrahrh qdpskecaga kgerrarhrg 961 gpragpreae sgeeparrhr arhkaqpahe aveketteke atekeaeive adkekelrnh 1021 qprephcdle tsgtvtvgpm htlpstclqk veeqpedadn qrnvtrmgsq ppdpntivhi 1081 pvmltgplge atvvpsgnvd lesqaegkke veaddvmrsg prpivpyssm fclsptnllr 1141 rfchyivtmr yfevvilvvi alssialaae dpvrtdsprn nalkyldyif tgvftfemvi 1201 kmidlglllh pgayfrdlwn ildfivvsga lvafafsgsk gkdintiksl rvlrvlrplk 1261 tikrlpklka vfdcvvnslk nvlnilivym lfmfifavia vqlfkgkffy ctdeskeler 1321 dcrgqyldye keeveaqprq wkkydfhydn vlwalltlft vstgegwpmv lkhsvdatye 1381 eqgpspgyrm elsifyvvyf vvfpfffvni fvaliiitfq eqgdkvmsec slekneraci 1441 dfaisakplt rympqnrqsf qyktwtfvvs ppfeyfimam ialntvvlmm kfydapyeye 1501 lmlkclnivf tsmfsmecvl kiiafgvlny frdawnvfdf vtvlgsitdi lvteiaetnn 1561 finlsflrlf raarlikllr qgytirillw tfvqsfkalp yvclliamlf fiyaiigmqv 1621 fgnialdddt sinrhnnfrt flqalmllfr satgeawhei mlsclsnqac deqanatecg 1681 sdfayfyfvs fiflcsflml nlfvavimdn feyltrdssi lgphhldefi rvwaeydpaa 1741 cgrisyndmf emlkhmsppl glgkkcparv aykrlvrmnm pisnedmtvh ftstlmalir 1801 taleiklapa gtkqhqcdae lrkeisvvwa nlpqktldll vpphkpdemt vgkvyaalmi 1861 fdfykqnktt rdqmqqapgg lsqmgpvslf hplkatleqt qpavlrgarv flrqksstsl 1921 snggaiqnqe sgikesvswg tqrtqdaphe arpplerghs teipvgrsga lavdvqmqsi 1981 trrgpdgepq pglesqgraa smprlaaetq pvtdaspmkr sistlaqrpr gthlcsttpd 2041 rpppsqassh hhhhrchrrr drkqrslekg pslsadmdga pssavgpglp pgegptgcrr 2101 ererrqergr sqerrqpsss ssekqrfysc drfggreppk pkpslsshpt sptagqepgp 2161 hpqgsgsvng spllstsgas tpgrggrrql pqtpltprps ityktanssp ihfagaqtsl 2221 pafspgrlsr glsehnallq rdplsqplap gsrigsdpyl gqrldseasv halpedtltf 2281 eeavatnsgr ssrtsyvssl tsqshplrrv pngyhctlgl ssggrarhsy hhpdqdhwc';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 maagcllalt ltlfqsllig psseepfpsa vtikswvdkm qedlvtlakt asgvnqlvdi 61 yekyqdlytv epnnarqlve iaardiekll snrskalvrl aleaekvqaa hqwredfasn 121 evvyynakdd ldpekndsep gsqrikpvfi edanfgrqis yqhaavhipt diyegstivl 181 nelnwtsald evfkknreed psllwqvfgs atglaryypa spwvdnsrtp nkidlydvrr 241 rpwyiqgaas pkdmlilvdv sgsvsgltlk lirtsvseml etlsdddfvn vasfnsnaqd 301 vscfqhlvqa nvrnkkvlkd avnnitakgi tdykkgfsfa feqllnynvs rancnkiiml 361 ftdggeeraq eifnkynkdk kvrvftfsvg qhnydrgpiq wmacenkgyy yeipsigair 421 intqeyldvl grpmvlagdk akqvqwtnvy ldalelglvi tgtlpvfnit gqfenktnlk 481 nqlilgvmgv dvsledikrl tprftlcpng yyfaidpngy vllhpnlqpk npksqepvtl 541 dfldaelend ikveirnkmi dgesgektfr tlvksqdery idkgnrtytw tpvngtdysl 601 alvlptysfy yikakleeti tqarskkgkm kdsetlkpdn feesgytfia prdycndlki 661 sdnnteflln fnefidrktp nnpscnadli nrvlldagft nelvqnywsk qknikgvkar 721 fvvtdggitr vypkeagenw qenpetyeds fykrsldndn yvftapyfnk sgpgayesgi 781 mvskaveiyi qgkllkpavv gikidvnswi enftktsird pcagpvcdck rnsdvmdcvi 841 lddggfllma nhddytnqig rffgeidpsl mrhlvnisvy afnksydyqs vcepgaapkq 901 gaghrsayvp svadilqigw wataaawsil qqfllsltfp rlleavemed ddftaslskq 961 sciteqtqyf fdndsksfsg vldcgncsri fhgeklmntn lifimveskg tcpcdtrlli 1021 qaeqtsdgpn pcdmvkqpry rkgpdvcfdn nvledytdcg gvsglnpslw yiigiqflll 1081 wlvsgsthrl l';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mavpartcga srpgpartar pwpgcgphpg pgtrrptsgp prplwlllpl lpllaapgas 61 aysfpqqhtm qhwarrleqe vdgvmrifgg vqqlreiykd nrnlfevqen epqklvekva 121 gdieslldrk vqalkrlada aenfqkahrw qdnikeediv yydakadael ddpesedver 181 gskastlrld fiedpnfknk vnysyaavqi ptdiykgstv ilnelnwtea lenvfmenrr 241 qdptllwqvf gsatgvtryy patpwrapkk idlydvrrrp wyiqgasspk dmviivdvsg 301 svsgltlklm ktsvcemldt lsdddyvnva sfnekaqpvs cfthlvqanv rnkkvfkeav 361 qgmvakgttg ykagfeyafd qlqnsnitra ncnkmimmft dggedrvqdv fekynwpnrt 421 vrvftfsvgq hnydvtplqw macankgyyf eipsigairi ntqeyldvlg rpmvlagkea 481 kqvqwtnvye dalglglvvt gtlpvfnltq dgpgekknql ilgvmgidva lndikrltpn 541 ytlgangyvf aidlngyvll hpnlkpqttn frepvtldfl daeledenke eirrsmidgn 601 kghkqirtlv kslderyide vtrnytwvpi rstnyslglv lppystfylq anlsdqilqv 661 kyfefllpss feseghvfia preyckdlna sdnnteflkn fielmekvtp dskqcnnfll 721 hnlildtgit qqlvervwrd qdlntyslla vfaatdggit rvfpnkaaed wtenpepfna 781 sfyrrsldnh gyvfkpphqd allrplelen dtvgilvsta velslgrrtl rpavvgvkld 841 leawaekfkv lasnrthqdq pqkcgpnshc emdcevnned llcvliddgg flvlsnqnhq 901 wdqvgrffse vdanlmlaly nnsfytrkes ydyqaacapq ppgnlgaapr gvfvptvadf 961 lnlawwtsaa awslfqqlly gliyhswfqa dpaeaegspe tresscvmkq tqyyfgsvna 1021 synaiidcgn csrlfhaqrl tntnllfvva ekplcsqcea grllqkethc padgpeqcel 1081 vqrpryrrgp hicfdynate dtsdcgrgas fppslgvlvs lqlllllglp prpqpqvlvh 1141 asrrl';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mvqktsmsrg pyppsqeipm evfdpspqgk yskrkgrfkr sdgstssdtt snsfvrqgsa 61 esytsrpsds dvsleedrea lrkeaerqal aqlekaktkp vafavrtnvg ynpspgdevp 121 vqgvaitfep kdflhikeky nndwwigrlv kegcevgfip spvkldslrl lqeqklrqnr 181 lgssksgdns ssslgdvvtg trrptppasa kqkqkstehv ppydvvpsmr piilvgpslk 241 gyevtdmmqk alfdflkhrf dgrisitrvt adislakrsv lnnpskhiii ersntrssla 301 evqseierif elartlqlva ldadtinhpa qlsktslapi ivyikitspk vlqrliksrg 361 ksqskhlnvq iaaseklaqc ppemfdiild enqledaceh laeyleaywk athppsstpp 421 npllnrtmat aalaaspapv snlqgpylas gdqpleratg ehasmheypg elgqppglyp 481 sshppgragt lralsrqdtf dadtpgsrns aytelgdscv dmetdpsegp glgdpagggt 541 pparqgswed eeedyeeelt dnrnrgrnka rycaegggpv lgrnkneleg wgrgvyir';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mqccglvhrr rvrvsygsad sytsrpsdsd vsleedreav rreaerqaqa qlekaktkpv 61 afavrtnvsy saaheddvpv pgmaisfeak dflhvkekfn ndwwigrlvk egceigfips 121 pvklenmrlq heqrakqgkf yssksggnss sslgdivpss rkstppssai didatgldae 181 endipanhrs pkpsansvts phskekrmpf fkktehtppy dvvpsmrpvv lvgpslkgye 241 vtdmmqkalf dflkhrfegr isitrvtadi slakrsvlnn pskhaiiers ntrsslaevq 301 seierifela rtlqlvvlda dtinhpaqls ktslapiivy vkisspkvlq rliksrgksq 361 akhlnvqmva adklaqcppe lfdvildenq ledacehlad yleaywkath ppssslpnpl 421 lsrtlatssl plsptlasns qgsqgdqrtd rsapirsasq aeeepsvepv kksqhrssss 481 aphhnhrsgt srglsrqetf dsetqesrds ayvepkedys hdhvdhyash rdhnhrdeth 541 gssdhrhres rhrsrdvdre qdhnecnkqr srhkskdryc ekdgeviskk rneagewnrd 601 vyirq';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 myddsyvpgf edseagsads ytsrpsldsd vsleedresa rrevesqaqq qlerakhkpv 61 afavrtnvsy cgvldeecpv qgsgvnfeak dflhikekys ndwwigrlvk eggdiafips 121 pqrlesirlk qeqkarrsgn psslsdignr rspppslakq kqkqaehvpp ydvvpsmrpv 181 vlvgpslkgy evtdmmqkal fdflkhrfdg risitrvtad lslakrsvln npgkrtiier 241 ssarssiaev qseierifel akslqlvvld adtinhpaql aktslapiiv fvkvsspkvl 301 qrlirsrgks qmkhltvqmm aydklvqcpp esfdvilden qledacehla eylevywrat 361 hhpapgpgll gppsaipglq nqqllgerge ehsplerdsl mpsdeasess rqawtgssqr 421 ssrhleedya dayqdlyqph rqhtsglpsa nghdpqdrll aqdsehnhsd rnwqrnrpwp 481 kdsy';

protein();

} else {

peptidesequence = '1 mydnlylhgi edseagsads ytsrpsdsdv sleedreair qereqqaaiq lerakskpva 61 favktnvsyc galdedvpvp staisfdakd flhikekynn dwwigrlvke gceigfipsp 121 lrleniriqq eqkrgrfhgg kssgnssssl gemvsgtfra tptstakqkq kvtehippyd 181 vvpsmrpvvl vgpslkgyev tdmmqkalfd flkhrfdgri sitrvtadis lakrsvlnnp 241 skraiiersn trsslaevqs eierifelar slqlvvldad tinhpaqlik tslapiivhv 301 kvsspkvlqr liksrgksqs khlnvqlvaa dklaqcppem fdvildenql edacehlgey 361 leaywratht tsstpmtpll grnlgstals pyptaisglq sqrmrhsnhs tenspierrs 421 lmtsdenyhn erarksrnrl ssssqhsrdh yplveedypd syqdtykphr nrgspggysh 481 dsrhrl';

protein();

}

} else if (randomizer <= 20) {

peptidesequence = '1 mtksngeepk mggrmerfqq gvrkrtllak kkvqnitked vksylfrnaf vlltvtaviv 61 gtilgftlrp yrmsyrevky fsfpgellmr mlqmlvlpli isslvtgmaa ldskasgkmg 121 mravvyymtt tiiavvigii iviiihpgkg tkenmhregk ivrvtaadaf ldlirnmfpp 181 nlveacfkqf ktnyekrsfk vpiqanetlv gavinnvsea metltritee lvpvpgsvng 241 vnalglvvfs mcfgfvignm keqgqalref fdslneaimr lvavimwyap vgilfliagk 301 ivemedmgvi ggqlamytvt vivgllihav ivlpllyflv trknpwvfig gllqalital 361 gtssssatlp itfkcleenn gvdkrvtrfv lpvgatinmd gtalyealaa ifiaqvnnfe 421 lnfgqiitis itataasiga agipqaglvt mvivltsvgl ptdditliia vdwfldrlrt 481 ttnvlgdslg agivehlsrh elknrdvemg nsvieenemk kpyqliaqdn etekpidset 541 km';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 maedadmrne leemqrradq ladeslestr rmlqlveesk dagirtlvml deqgeqldrv 61 eegmnhinqd mkeaeknlkd lgkccglfic pcnklkssda ykkawgnnqd gvvasqparv 121 vdereqmais ggfirrvtnd arenemdenl eqvsgiignl rhmaldmgne idtqnrqidr 181 imekadsnkt rideanqrat kmlgsg';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkdrtqelrt akdsdddddv avtvdrdrfm deffeqveei rgfidkiaen veevkrkhsa 61 ilaspnpdek tkeeleelms dikktankvr sklksieqsi eqeeglnrss adlrirktqh 121 stlsrkfvev mseynatqsd yrerckgriq rqleitgrtt tseeledmle sgnpaifasg 181 iimdssiskq alseietrhs eiiklensir elhdmfmdma mlvesqgemi drieynveha 241 vdyveravsd tkkavkyqsk arrkkimiii ccvilgivia stvggifa';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 mklyslsvly kgeakvvllk aaydvssfsf fqrssvqefm tftsqliver sskgtrasvk 61 eqdylchvyv rndslagvvi adneypsrva ftllekvlde fskqvdridw pvgspatihy 121 paldghlsry qnpreadpmt kvqaeldetk iilhntmesl lergeklddl vsksevlgtq 181 skafyktark qnsccaim';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 magrsmqaar cptdelsltn cavvnekdfq sgqhvivrts pnhrytftlk thpsvvpgsi 61 afslpqrkwa glsigqeiev slytfdkakq cigtmtieid flqkksidsn pydtdkmaae 121 fiqqfnnqaf svgqqlvfsf neklfgllvk dieamdpsil kgepatgkrq kievglvvgn 181 sqvafekaen sslnligkak tkenrqsiin pdwnfekmgi ggldkefsdi frrafasrvf 241 ppeiveqmgc khvkgillyg ppgcgktlla rqigkmlnar epkvvngpei lnkyvgesea 301 nirklfadae eeqrrlgans glhiiifdei daickqrgsm agstgvhdtv vnqllskidg 361 veqlnnilvi gmtnrpdlid eallrpgrle vkmeiglpde kgrlqilhih tarmrghqll 421 sadvdikela vetknfsgae leglvraaqs tamnrhikas tkvevdmeka eslqvtrgdf 481 laslendikp afgtnqedya syimngiikw gdpvtrvldd gellvqqtkn sdrtplvsvl 541 legpphsgkt alaakiaees nfpfikicsp dkmigfseta kcqamkkifd dayksqlscv 601 vvddierlld yvpigprfsn lvlqallvll kkappqgrkl liigttsrkd vlqememlna 661 fsttihvpni atgeqlleal ellgnfkdke rttiaqqvkg kkvwigikkl lmliemslqm 721 dpeyrvrkfl allreegasp ldfd';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 masatdsryg qkessdqnfd ymfkiliign ssvgktsflf ryaddsftpa fvstvgidfk 61 vktiyrndkr iklqiwdtag qeryrtitta yyrgamgfil myditneesf navqdwstqi 121 ktyswdnaqv llvgnkcdme dervvsserg rqladhlgfe ffeasakdni nvkqtferlv 181 dvicekmses ldtadpavtg akqgpqlsdq qvpphqdcac';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 msllcvrvkr akfqgspdkf ntyvtlkvqn vksttvavrg dqpsweqdfm feisrldlgl 61 svevwnkgli wdtmvgtvwi alktirqsde egpgewstle aetlmkddei cgtrnptphk 121 illdtrfelp fdipeeeary wtykweqina lgadneyssq eesqrkplpt aaaqcsfedp 181 dsavddrdsd yrsetsnsfp ppyhtasqpn asvhqfpvpv rspqqlllqg ssrdscndsm 241 qsydldyper raisptsssr ygsscnvsqg ssqlseldqy heqdddhret dsihschssh 301 slsrdgqagf geqekplevt gqaekeaace pkemkedatt hpppdlvlqk dhflgpqesf 361 peenasspft qarahwirav tkvrlqlqei pddgdpslpq wlpegpaggl ygidsmpdlr 421 rkkplplvsd lslvqsrkag itsamatrts lkdeelkshv ykktlqaliy piscttphnf 481 evwtattpty cyecegllwg iarqgmrcse cgvkchekcq dllnadclqr aaeksckhga 541 edrtqniima mkdrmkirer nkpeifevir dvftvnkaah vqqmktvkqs vldgtskwsa 601 kititvvcaq glqakdktgs sdpyvtvqvs ktkkrtktif gnlnpvweek fhfechnssd 661 rikvrvwded ddiksrvkqr lkresddflg qtiievrtls gemdvwynle krtdksavsg 721 airlqisvei kgeekvapyh vqytclhenl fhyltdiqgs ggvripearg ddawkvyfde 781 taqeivdefa mrygiesiyq amthfaclss kymcpgvpav mstllanina yyahttastn 841 vsasdrfaas nfgkerfvkl ldqlhnslri dlstyrnnfp agsperlqdl kstvdlltsi 901 tffrmkvqel qspprasqvv kdcvkaclns tyeyifnnch dlysrqyqlk qelppeeqgp 961 sirnldfwpk litlivsiie edknsytpvl nqfpqelnvg kvsaevmwhl faqdmkyale 1021 ehekdhlcks adymnlhfkv kwlhneyvrd lpvlqgqvpe ypawfeqfvl qwldenedvs 1081 leflrgaler dkkdgfqqts ehalfscsvv dvftqlnqsf eiirklecpd psilahymrr 1141 faktigkvlm qyadilskdf payctkeklp cilmnnvqql rvqlekmfea mggkeldlea 1201 adslkelqvk lntvldelsm vfgnsfqvri decvrqmadi lgqvrgtgna spdarasaaq 1261 dadsvlrplm dfldgnltlf atvcektvlk rvlkelwrvv mntmermivl ppltdqtgtq 1321 liftaakels hlsklkdhmv reetrnltpk qcavldlald tikqyfhagg nglkktflek 1381 spdlqslrya lslytqttdt liktfvrsqt tqgsgvddpv gevsiqvdlf thpgtgehkv 1441 tvkvvaandl kwqtagmfrp fvevtmvgph qsdkkrkftt ksksnnwapk ynetfhfllg 1501 neegpesyel qicvkdycfa redrvlglav mplrdvtakg scacwcplgr kihmdetglt 1561 ilrilsqrsn devarefvkl ksesrsteeg s';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mapiglkavv gekimhdvik kvkkkgewkv lvvdqlsmrm lsscckmtdi mtegitived 61 inkrreplps leavylitps eksvhslisd fkdpptakyr aahvfftdsc pdalfnelvk 121 sraakviktl teiniaflpy esqvysldsa dsfqsfysph kaqmknpile rlaeqiatlc 181 atlkeypavr yrgeykdnal laqliqdkld aykaddptmg egpdkarsql lildrgfdps 241 spvlheltfq amsydllpie ndvykyetsg igearvkevl ldedddlwia lrhkhiaevs 301 qevtrslkdf ssskrmntge kttmrdlsqm lkkmpqyqke lskysthlhl aedcmkhyqg 361 tvdklcrveq dlamgtdaeg ekikdpmrai vpilldanvs tydkiriill yiflkngite 421 enlnkliqha qippedseii tnmahlgvpi vtdstlrrrs kperkerise qtyqlsrwtp 481 iikdimedti edkldtkhyp yistrssasf sttavsaryg hwhknkapge yrsgprliif 541 ilggvslnem rcayevtqan gkwevligst hiltptkflm dlrhpdfres srvsfedqap 601 tme';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mpavskgdgm rglavfisdi rnckskeaei krinkelani rskfkgdkal dgyskkkyvc 61 kllfifllgh didfghmeav nllssnryte kqigylfisv lvnsnselir linnaikndl 121 asrnptfmgl alhciasvgs remaeafage ipkvlvagdt mdsvkqsaal cllrlyrtsp 181 dlvpmgdwts rvvhllndqh lgvvtaatsl ittlaqknpe efktsvslav srlsrivtsa 241 stdlqdytyy fvpapwlsvk llrllqcypp pedpavrgrl tecletilnk aqeppkskkv 301 qhsnaknavl feaisliihh dsepnllvra cnqlgqflqh retnlrylal esmctlasse 361 fsheavkthi etvinalkte rdvsvrqrav dllyamcdrs napqivaeml syletadysi 421 reeivlkvai laekyavdyt wyvdtilnli riagdyvsee vwyrviqivi nrddvqgyaa 481 ktvfealqap achenlvkvg gyilgefgnl iagdprsspl iqfhllhskf hlcsvptral 541 llstyikfvn lfpevkptiq dvlrsdsqlr nadvelqqra veylrlstva stdilatvle 601 emppfperes silaklkkkk gpstvtdled tkrdrsvdvn ggpepapast savstpspsa 661 dllglgaapp apagpppssg gsgllvdvfs dsasvvapla pgsednfarf vcknngvlfe 721 nqllqiglks efrqnlgrmf ifygnktstq flnftptlic sddlqpnlnl qtkpvdptve 781 ggaqvqqvvn iecvsdftea pvlniqfryg gtfqnvsvql pitlnkffqp temasqdffq 841 rwkqlsnpqq evqnifkakh pmdtevtkak iigfgsalle evdpnpanfv gagiihtktt 901 qigcllrlep nlqaqmyrlt lrtskeavsq rlcellsaqf';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 msgqtltdri aaaqysvtgs avaravckat thevmgpkkk hldyliqatn etnvnipqma 61 dtlferatns swvvvfkalv tthhlmvhgn erfiqylasr ntlfnlsnfl dksgshgydm 121 stfirrysry lnekafsyrq mafdfarvkk gadgvmrtma pekllksmpi lqgqidalle 181 fdvhpneltn gvinaafmll fkdliklfac yndgvinlle kffemkkgqc kdaleiykrf 241 ltrmtrvsef lkvaeqvgid kgdipdltqa psslmetleq hlntlegkkp gnnegsgaps 301 plsksspatt vtspnstpak tidtsppvdl fatasaavpv stskpssdll dlqpdfssgg 361 aaaaaapapp ppaggatawg dllgedslaa lssvpseaqi sdpfapeptp ptttaeiata 421 sasasttttv tavtaevdlf gdafaaspge apaasegaaa patptpvaaa ldacsgndpf 481 apsegsaeaa peldlfamkp petsvpvvtp tastappvpa tapspapava aaaaattaat 541 aaatttttts aatattappa ldifgdlfes tpevaaapkp daapsidlfs tdafssppqg 601 aspvpesslt adllsvdafa apspattasp akvdssgvid lfgdafgssa sepqpasqaa 661 ssssasadll agfggsfmap spspvtpaqn nllqpnfeaa fgttpstsss ssfdpsvfdg 721 lgdllmptma pagqpapvsm vppspamaas kalgsdldss laslvgnlgi sgtttkkgdl 781 qwnagekklt gganwqpkva patwsagvpp saplqgavpp tssvppvaga psvgqpgagf 841 gmppagtgmp mmpqqpvmfa qpmmrppfga aavpgtqlsp sptpasqspk kppakdplad 901 lnikdfl';

protein();

} else if (randomizer <= 120) {

peptidesequence = '1 maeldpfgap agapggpalg ngvagageed paaaflaqqe seiagiende afaildggap 61 gpqphgeppg gpdavdgvmn geyyqesngp tdsyaaisqv drlqsepesi rkwreeqmer 121 lealdansrk qeaewkekai keleewyarq deqlqktkan nraaeeafvn didesspgte 181 wervarlcdf npksskqakd vsrmrsvlis lkqaplvh';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 matidgrpaq ygislkqlre lmehrgregv mkiaenggih elckklytsp neglsgskad 61 eehrretfgs nvippkppkt fltlvwealq dvtliileva alvslglsfy kpadedapvl 121 qeeeehhgwi eglailisvi vvvivtafnd yskerqfrgl qnriegehkf svirggevcq 181 isvgdilvgd iaqvkygdll padgcliqsn dlkvdesslt gesdhvkkgp dvdpmvlsgt 241 hvmegsgkmv vtavgvnsqa giiftllgaa vdeqeaeikk mkkeakrank qknltgendg 301 rsqikgsqap sqretvtsei tksesegnhl pqssssgaae tghkkeksvl qakltklaiq 361 igyagstiav ltviiliiqf ciktfvidek pwkntyannl vkhliigvtv lvvavpeglp 421 lavtlslays vkkmmkdnnl vrhldacetm gnataicsdk tgtlttnrmt vvqsyicekl 481 ckvlptlsdi pqhvgnlitm gisvnsayts nimaghnpgd lpiqvgnkte callgfvqgl 541 gvkyqsirde itedkftrvy tfnsvrksmg tviprpnggy rlytkgasei imkkcafiyg 601 hegtlekftr dmqerlirev iepmacdglr tisvayrdfv pgkaainevh idgepnwdde 661 enimtnltcl cvvgiedpvr pevpdairkc qragitvrmv tgdnintars iaskcgilrp 721 nddflilegk efnrrirdsn gdiqqhlidk vwpklrvlar ssptdkytlv kgiidstvse 781 nrevvavtgd gtndgpalkk advgfamgia gtdvakeasd iiltddnfss ivkavmwgrn 841 vydsiakflq fqltvnvvav ivafigacav qdsplkavqm lwvnlimdtl aslalatefp 901 tpdlllrkpy grtkplisrt mmknilgqal yqliiifgll fvgdvildie sgrgqelnag 961 ptqhftiifn tfvmmtlfne inarkihgqr nvieglltnp ifytiwiftm isqvliiqyg 1021 kmafstkalt ldqwlwciff gigtlvwgql itsvptrklp kilswgrghp eeytdgmnlg 1081 eerfdsidsd kkpragqilw irgltrlqtq viggelqerl ipvpysksnt dqairvvnaf 1141 rqgldarygd htntslaevl rkqtslskrl setssieyad nipdeltipe idverlsshs 1201 htetav';

protein();

} else {

peptidesequence = '1 maqilpirfq ehlqlqnlgi npanigfstl tmesdkfici rekvgeqaqv viidmndpsn 61 pirrpisads aimnpaskvi alkagktlqi fniemkskmk ahtmtddvtf wkwislntva 121 lvtdnavyhw smegesqpvk mfdrhsslag cqiinyrtda kqkwllltgi saqqnrvvga 181 mqlysvdrkv sqpieghaas faqfkmegna eestlfcfav rgqaggklhi ievgtpptgn 241 qpfpkkavdv ffppeaqndf pvamqisekh dvvflitkyg yihlydletg tciymnrisg 301 etifvtaphe atagiigvnr kgqvlsvcve eeniipyitn vlqnpdlalr mavrnnlaga 361 eelfarkfna lfaqgnysea akvaanapkg ilrtpdtirr fqsvpaqpgq tspllqyfgi 421 lldqgqlnky eslelcrpvl qqgrkqllek wlkedklecs eelgdlvksv dptlalsvyl 481 ranvpnkviq cfaetgqvqk ivlyakkvgy tpdwifllrn vmrispdqgq qfaqmlvqde 541 epladitqiv dvfmeynliq qctaflldal knnrpsegpl qtrllemnlm hapqvadail 601 gnqmfthydr ahiaqlceka gllqralehf tdlydikrav vhthllnpew lvnyfgslsv 661 edsleclram lsanirqnlq icvqvaskyh eqlstqslie lfesfksfeg lfyflgsivn 721 fsqdpdvhfk yiqaacktgq ikevericre sncydpervk nflkeakltd qlpliivcdr 781 fdfvhdlvly lyrnnlqkyi eiyvqkvnps rlpvviggll dvdcsedvik nlilvvrgqf 841 stdelvaeve krnrlklllp wlearihegc eepathnala kiyidsnnnp erflrenpyy 901 dsrvvgkyce krdphlacva yergqcdlel invcnenslf kslsrylvrr kdpelwgsvl 961 lesnpyrrpl idqvvqtals etqdpeevsv tvkafmtadl pneliellek ivldnsvfse 1021 hrnlqnllil taikadrtrv meyinrldny dapdianiai snelfeeafa ifrkfdvnts 1081 avqvliehig nldrayefae rcnepavwsq lakaqlqkgm vkeaidsyik addpssymev 1141 vqaantsgnw eelvkylqma rkkaresyve telifalakt nrlaeleefi ngpnnahiqq 1201 vgdrcydekm ydaakllynn vsnfgrlast lvhlgeyqaa vdgarkanst rtwkevcfac 1261 vdgkefrlaq mcglhivvha deleelinyy qdrgyfeeli tmleaalgle rahmgmftel 1321 ailyskfkpq kmrehlelfw srvnipkvlr aaeqahlwae lvflydkyee ydnaiitmmn 1381 hptdawkegq fkdiitkvan velyyraiqf ylefkpllln dllmvlsprl dhtravnyfs 1441 kvkqlplvkp ylrsvqnhnn ksvneslnnl fiteedyqal rtsidaydnf dnislaqrle 1501 kheliefrri aaylfkgnnr wkqsvelckk dslykdamqy aseskdtela eellqwflqe 1561 ekrecfgacl ftcydllrpd vvletawrhn imdfampyfi qvmkeyltkv dkldaseslr 1621 keeeqatetq pivygqpqlm ltagpsvavp pqapfgygyt appygqpqpg fgysm';

protein();

}

}

function synapse\_x() {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x + 6.187927353e+27;

cursor1z = cursor2z;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mehgtllaqp glwtrdtswa llyflcyilp qtapqvlrig gifetvenep vnveelafkf 61 avtsinrnrt lmpnttltyd iqrinlfdsf easrracdql algvaalfgp shsssvsavq 121 sicnalevph iqtrwkhpsv dnkdlfyinl ypdyaaisra ildlvlyynw ktvtvvyeds 181 tglirlqeli kapsryniki kirqlpsgnk dakpllkemk kgkefyvifd cshetaaeil 241 kqilfmgmmt eyyhyffttl dlfaldlely rysgvnmtgf rllnidnphv ssiiekwsme 301 rlqapprpet glldgmmtte aalmydavym vaiashrasq ltvsslqchr hkpwrlgprf 361 mnlikearwd gltghitfnk tnglrkdfdl diislkeegt ekaagevskh lykvwkkigi 421 wnsnsglnmt dsnkdkssni tdslanrtli vttileepyv myrksdkply gndrfegycl 481 dllkelsnil gfiydvklvp dgkygaqndk gewngmvkel idhradlava pltityvrek 541 vidfskpfmt lgisilyrkp ngtnpgvfsf lnplspdiwm yvllaclgvs cvlfviarft 601 pyewynphpc npdsdvvenn ftllnsfwfg vgalmqqgse lmpkalstri vggiwwfftl 661 iiissytanl aafltverme spidsaddla kqtkieygav rdgstmtffk kskistyekm 721 wafmssrqqt alvrnsdegi qrvlttdyal lmestsieyv tqrncnltqi gglidskgyg 781 vgtpigspyr dkitiailql qeegklhmmk ekwwrgngcp eednkeasal gveniggifi 841 vlaaglvlsv fvaigefiyk srknndieqa fcffyglqck qthptnstsg ttlstdlecg 901 klireergir kqssvhtv';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkiifpilsn pvfrrtvkll lcllwigysq gtthvlrfgg ifeyvesgpm gaeelafrfa 61 vntinrnrtl lpnttltydt qkinlydsfe askkacdqls lgvaaifgps hsssanavqs 121 icnalgvphi qtrwkhqvsd nkdsfyvsly pdfsslsrai ldlvqffkwk tvtvvyddst 181 glirlqelik apsrynlrlk irqlpadtkd akpllkemkr gkefhvifdc shemaagilk 241 qalamgmmte yyhyifttld lfaldvepyr ysgvnmtgfr ilntentqvs siiekwsmer 301 lqappkpdsg lldgfmttda almydavhvv svavqqfpqm tvsslqcnrh kpwrfgtrfm 361 slikeahweg ltgritfnkt nglrtdfdld vislkeegle kigtwdpasg lnmtesqkgk 421 panitdslsn rslivttile epyvlfkksd kplygndrfe gycidllrel stilgftyei 481 rlvedgkyga qddangqwng mvrelidhka dlavaplait yvrekvidfs kpfmtlgisi 541 lyrkpngtnp gvfsflnpls pdiwmyilla ylgvscvlfv iarfspyewy nphpcnpdsd 601 vvennftlln sfwfgvgalm qqgselmpka lstrivggiw wfftliiiss ytanlaaflt 661 vermespids addlakqtki eygavedgat mtffkkskis tydkmwafms srrqsvlvks 721 neegiqrvlt sdyaflmest tiefvtqrnc nltqigglid skgygvgtpm gspyrdkiti 781 ailqlqeegk lhmmkekwwr gngcpeeesk easalgvqni ggifivlaag lvlsvfvavg 841 eflykskkna qlekrsfcsa mveelrmslk cqrrlkhkpq apvivkteev inmhtfndrr 901 lpgketma';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mtapwrrlrs lvweywagll vcafwipdsr gmphvirigg ifeyadgpna qvmnaeehaf 61 rfsaniinrn rtllpnttlt ydiqrihfhd sfeatkkacd qlalgvvaif gpsqgsctna 121 vqsicnalev phiqlrwkhh pldnkdtfyv nlypdyasls haildlvqyl kwrsatvvyd 181 dstglirlqe limapsryni rlkirqlpid sddsrpllke mkrgrefrii fdcshtmaaq 241 ilkqamamgm mteyyhfift tldlyaldle pyrysgvnlt gfrilnvdnp hvsaivekws 301 merlqaaprs esglldgvmm tdaallydav hivsvcyqra pqmtvnslqc hrhkawrfgg 361 rfmnfikeaq wegltgrivf nktsglrtdf dldiislked glekvgvwsp adglniteva 421 kgrgpnvtds ltnrslivtt vleepfvmfr ksdrtlygnd rfegycidll kelahilgfs 481 yeirlvedgk ygaqddkgqw ngmvkelidh kadlavaplt ithvrekaid fskpfmtlgv 541 silyrkpngt npsvfsflnp lspdiwmyvl laylgvscvl fviarfspye wydahpcnpg 601 sevvennftl lnsfwfgmgs lmqqgselmp kalstriigg iwwfftliii ssytanlaaf 661 ltvermespi dsaddlakqt kieygavkdg atmtffkksk istfekmwaf msskpsalvk 721 nneegiqral tadyallmes ttieyvtqrn cnltqiggli dskgygigtp mgspyrdkit 781 iailqlqeed klhimkekww rgsgcpeeen keasalgiqk iggifivlaa glvlsvlvav 841 gefvyklrkt aereqrsfcs tvadeirfsl tcqrrvkhkp qppmmvktda vinmhtfndr 901 rlpgkdsmac stslapvfp';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mprvsaplvl lpawlvmvac sphslriaai lddpmecsrg erlsitlakn rinraperlg 61 kakvevdife llrdseyeta etmcqilpkg vvavlgpsss passsiisni cgekevphfk 121 vapeefvkfq fqrfttlnlh psntdisvav agilnffnct taclicakae cllnlekllr 181 qfliskdtls vrmlddtrdp tpllkeirdd ktatiiihan asmshtillk aaelgmvsay 241 ytyiftnlef slqrmdslvd drvnilgfsi fnqshaffqe faqslnqswq encdhvpftg 301 palssallfd avyavvtavq elnrsqeigv kplscgsaqi wqhgtslmny lrmveleglt 361 ghiefnskgq rsnyalkilq ftrngfrqig qwhvaeglsm dshlyasnis dtlfnttlvv 421 ttilenpylm lkgnhqemeg ndryegfcvd mlkelaeilr fnykirlvgd gvygvpeang 481 twtgmvgeli arkadlavag ltitaerekv idfskpfmtl gisilyrvhm grkpgyfsfl 541 dpfspgvwlf mllaylavsc vlflvarltp yewysphpca qgrcnllvnq yslgnslwfp 601 vggfmqqgst iapralstrc vsgvwwaftl iiissytanl aafltvqrmd vpiesvddla 661 dqtaieygti hggssmtffq nsryqtyqrm wnymyskqps vfvksteegi arvlnsnyaf 721 llestmneyy rqrncnltqi gglldtkgyg igmpvgsvfr defdlailql qennrleilk 781 rkwweggkcp keedhrakgl gmeniggifv vlicglivai fmamleflwt lrhseatevs 841 vcqemvtelr siilcqdsih prrrraavpp prppipeerr prgtatlsng klcgagepdq 901 laqrlaqeaa lvargcthir vcpecrrfqg lrarpspars eeslewektt nssepe';

protein();

} else {

peptidesequence = '1 mpaellllli vafaspscqv lsslrmaail ddqtvcgrge rlalalareq ingiievpak 61 arvevdifel qrdsqyettd tmcqilpkgv vsvlgpsssp asastvshic gekeiphikv 121 gpeetprlqy lrfasvslyp snedvslavs rilksfnyps aslicakaec llrleelvrg 181 flisketlsv rmlddsrdpt pllkeirddk vstiiidana sishlilrka selgmtsafy 241 kyilttmdfp ilhldgived ssnilgfsmf ntshpfypef vrslnmswre nceastylgp 301 alsaalmfda vhvvvsavre lnrsqeigvk plactsaniw phgtslmnyl rmveydgltg 361 rvefnskgqr tnytlrilek srqghreigv wysnrtlamn attldinlsq tlanktlvvt 421 tilenpyvmr rpnfqalsgn erfegfcvdm lrelaellrf ryrlrlvedg lygapepngs 481 wtgmvgelin rkadlavaaf titaerekvi dfskpfmtlg isilyrvhmg rkpgyfsfld 541 pfspavwlfm llaylavscv lflaarlspy ewynphpclr arphilenqy tlgnslwfpv 601 ggfmqqgsei mpralstrcv sgvwwaftli iissytanla afltvqrmev pvesaddlad 661 qtnieygtih agstmtffqn sryqtyqrmw nymqskqpsv fvksteegia rvlnsryafl 721 lestmneyhr rlncnltqig glldtkgygi gmplgspfrd eitlailqlq ennrleilkr 781 kwweggrcpk eedhrakglg meniggifiv licgliiavf vavmefiwst rrsaeseetp 841 alhpaacqcs algprtplke psmllvkvps trvqvafsrt slrqvcpfll qhqlsslywi 901 qatnvqicch fsslkpspdl tfppshrpls sllftalaav gglpdassff fppisscppl 961 qsgigpchst eatlvtsnfh v';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x - 6.187927353e+27;

cursor1y = cursor2y;

cursor1z = cursor2z;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

function synapse\_y() {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y - 6.187927353e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y + 6.187927353e+27;

cursor1z = cursor2z;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mehgtllaqp glwtrdtswa llyflcyilp qtapqvlrig gifetvenep vnveelafkf 61 avtsinrnrt lmpnttltyd iqrinlfdsf easrracdql algvaalfgp shsssvsavq 121 sicnalevph iqtrwkhpsv dnkdlfyinl ypdyaaisra ildlvlyynw ktvtvvyeds 181 tglirlqeli kapsryniki kirqlpsgnk dakpllkemk kgkefyvifd cshetaaeil 241 kqilfmgmmt eyyhyffttl dlfaldlely rysgvnmtgf rllnidnphv ssiiekwsme 301 rlqapprpet glldgmmtte aalmydavym vaiashrasq ltvsslqchr hkpwrlgprf 361 mnlikearwd gltghitfnk tnglrkdfdl diislkeegt ekaagevskh lykvwkkigi 421 wnsnsglnmt dsnkdkssni tdslanrtli vttileepyv myrksdkply gndrfegycl 481 dllkelsnil gfiydvklvp dgkygaqndk gewngmvkel idhradlava pltityvrek 541 vidfskpfmt lgisilyrkp ngtnpgvfsf lnplspdiwm yvllaclgvs cvlfviarft 601 pyewynphpc npdsdvvenn ftllnsfwfg vgalmqqgse lmpkalstri vggiwwfftl 661 iiissytanl aafltverme spidsaddla kqtkieygav rdgstmtffk kskistyekm 721 wafmssrqqt alvrnsdegi qrvlttdyal lmestsieyv tqrncnltqi gglidskgyg 781 vgtpigspyr dkitiailql qeegklhmmk ekwwrgngcp eednkeasal gveniggifi 841 vlaaglvlsv fvaigefiyk srknndieqa fcffyglqck qthptnstsg ttlstdlecg 901 klireergir kqssvhtv';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkiifpilsn pvfrrtvkll lcllwigysq gtthvlrfgg ifeyvesgpm gaeelafrfa 61 vntinrnrtl lpnttltydt qkinlydsfe askkacdqls lgvaaifgps hsssanavqs 121 icnalgvphi qtrwkhqvsd nkdsfyvsly pdfsslsrai ldlvqffkwk tvtvvyddst 181 glirlqelik apsrynlrlk irqlpadtkd akpllkemkr gkefhvifdc shemaagilk 241 qalamgmmte yyhyifttld lfaldvepyr ysgvnmtgfr ilntentqvs siiekwsmer 301 lqappkpdsg lldgfmttda almydavhvv svavqqfpqm tvsslqcnrh kpwrfgtrfm 361 slikeahweg ltgritfnkt nglrtdfdld vislkeegle kigtwdpasg lnmtesqkgk 421 panitdslsn rslivttile epyvlfkksd kplygndrfe gycidllrel stilgftyei 481 rlvedgkyga qddangqwng mvrelidhka dlavaplait yvrekvidfs kpfmtlgisi 541 lyrkpngtnp gvfsflnpls pdiwmyilla ylgvscvlfv iarfspyewy nphpcnpdsd 601 vvennftlln sfwfgvgalm qqgselmpka lstrivggiw wfftliiiss ytanlaaflt 661 vermespids addlakqtki eygavedgat mtffkkskis tydkmwafms srrqsvlvks 721 neegiqrvlt sdyaflmest tiefvtqrnc nltqigglid skgygvgtpm gspyrdkiti 781 ailqlqeegk lhmmkekwwr gngcpeeesk easalgvqni ggifivlaag lvlsvfvavg 841 eflykskkna qlekrsfcsa mveelrmslk cqrrlkhkpq apvivkteev inmhtfndrr 901 lpgketma';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mtapwrrlrs lvweywagll vcafwipdsr gmphvirigg ifeyadgpna qvmnaeehaf 61 rfsaniinrn rtllpnttlt ydiqrihfhd sfeatkkacd qlalgvvaif gpsqgsctna 121 vqsicnalev phiqlrwkhh pldnkdtfyv nlypdyasls haildlvqyl kwrsatvvyd 181 dstglirlqe limapsryni rlkirqlpid sddsrpllke mkrgrefrii fdcshtmaaq 241 ilkqamamgm mteyyhfift tldlyaldle pyrysgvnlt gfrilnvdnp hvsaivekws 301 merlqaaprs esglldgvmm tdaallydav hivsvcyqra pqmtvnslqc hrhkawrfgg 361 rfmnfikeaq wegltgrivf nktsglrtdf dldiislked glekvgvwsp adglniteva 421 kgrgpnvtds ltnrslivtt vleepfvmfr ksdrtlygnd rfegycidll kelahilgfs 481 yeirlvedgk ygaqddkgqw ngmvkelidh kadlavaplt ithvrekaid fskpfmtlgv 541 silyrkpngt npsvfsflnp lspdiwmyvl laylgvscvl fviarfspye wydahpcnpg 601 sevvennftl lnsfwfgmgs lmqqgselmp kalstriigg iwwfftliii ssytanlaaf 661 ltvermespi dsaddlakqt kieygavkdg atmtffkksk istfekmwaf msskpsalvk 721 nneegiqral tadyallmes ttieyvtqrn cnltqiggli dskgygigtp mgspyrdkit 781 iailqlqeed klhimkekww rgsgcpeeen keasalgiqk iggifivlaa glvlsvlvav 841 gefvyklrkt aereqrsfcs tvadeirfsl tcqrrvkhkp qppmmvktda vinmhtfndr 901 rlpgkdsmac stslapvfp';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mprvsaplvl lpawlvmvac sphslriaai lddpmecsrg erlsitlakn rinraperlg 61 kakvevdife llrdseyeta etmcqilpkg vvavlgpsss passsiisni cgekevphfk 121 vapeefvkfq fqrfttlnlh psntdisvav agilnffnct taclicakae cllnlekllr 181 qfliskdtls vrmlddtrdp tpllkeirdd ktatiiihan asmshtillk aaelgmvsay 241 ytyiftnlef slqrmdslvd drvnilgfsi fnqshaffqe faqslnqswq encdhvpftg 301 palssallfd avyavvtavq elnrsqeigv kplscgsaqi wqhgtslmny lrmveleglt 361 ghiefnskgq rsnyalkilq ftrngfrqig qwhvaeglsm dshlyasnis dtlfnttlvv 421 ttilenpylm lkgnhqemeg ndryegfcvd mlkelaeilr fnykirlvgd gvygvpeang 481 twtgmvgeli arkadlavag ltitaerekv idfskpfmtl gisilyrvhm grkpgyfsfl 541 dpfspgvwlf mllaylavsc vlflvarltp yewysphpca qgrcnllvnq yslgnslwfp 601 vggfmqqgst iapralstrc vsgvwwaftl iiissytanl aafltvqrmd vpiesvddla 661 dqtaieygti hggssmtffq nsryqtyqrm wnymyskqps vfvksteegi arvlnsnyaf 721 llestmneyy rqrncnltqi gglldtkgyg igmpvgsvfr defdlailql qennrleilk 781 rkwweggkcp keedhrakgl gmeniggifv vlicglivai fmamleflwt lrhseatevs 841 vcqemvtelr siilcqdsih prrrraavpp prppipeerr prgtatlsng klcgagepdq 901 laqrlaqeaa lvargcthir vcpecrrfqg lrarpspars eeslewektt nssepe';

protein();

} else {

peptidesequence = '1 mpaellllli vafaspscqv lsslrmaail ddqtvcgrge rlalalareq ingiievpak 61 arvevdifel qrdsqyettd tmcqilpkgv vsvlgpsssp asastvshic gekeiphikv 121 gpeetprlqy lrfasvslyp snedvslavs rilksfnyps aslicakaec llrleelvrg 181 flisketlsv rmlddsrdpt pllkeirddk vstiiidana sishlilrka selgmtsafy 241 kyilttmdfp ilhldgived ssnilgfsmf ntshpfypef vrslnmswre nceastylgp 301 alsaalmfda vhvvvsavre lnrsqeigvk plactsaniw phgtslmnyl rmveydgltg 361 rvefnskgqr tnytlrilek srqghreigv wysnrtlamn attldinlsq tlanktlvvt 421 tilenpyvmr rpnfqalsgn erfegfcvdm lrelaellrf ryrlrlvedg lygapepngs 481 wtgmvgelin rkadlavaaf titaerekvi dfskpfmtlg isilyrvhmg rkpgyfsfld 541 pfspavwlfm llaylavscv lflaarlspy ewynphpclr arphilenqy tlgnslwfpv 601 ggfmqqgsei mpralstrcv sgvwwaftli iissytanla afltvqrmev pvesaddlad 661 qtnieygtih agstmtffqn sryqtyqrmw nymqskqpsv fvksteegia rvlnsryafl 721 lestmneyhr rlncnltqig glldtkgygi gmplgspfrd eitlailqlq ennrleilkr 781 kwweggrcpk eedhrakglg meniggifiv licgliiavf vavmefiwst rrsaeseetp 841 alhpaacqcs algprtplke psmllvkvps trvqvafsrt slrqvcpfll qhqlsslywi 901 qatnvqicch fsslkpspdl tfppshrpls sllftalaav gglpdassff fppisscppl 961 qsgigpchst eatlvtsnfh v';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

function synapse\_y2() {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y - 6.187927353e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mehgtllaqp glwtrdtswa llyflcyilp qtapqvlrig gifetvenep vnveelafkf 61 avtsinrnrt lmpnttltyd iqrinlfdsf easrracdql algvaalfgp shsssvsavq 121 sicnalevph iqtrwkhpsv dnkdlfyinl ypdyaaisra ildlvlyynw ktvtvvyeds 181 tglirlqeli kapsryniki kirqlpsgnk dakpllkemk kgkefyvifd cshetaaeil 241 kqilfmgmmt eyyhyffttl dlfaldlely rysgvnmtgf rllnidnphv ssiiekwsme 301 rlqapprpet glldgmmtte aalmydavym vaiashrasq ltvsslqchr hkpwrlgprf 361 mnlikearwd gltghitfnk tnglrkdfdl diislkeegt ekaagevskh lykvwkkigi 421 wnsnsglnmt dsnkdkssni tdslanrtli vttileepyv myrksdkply gndrfegycl 481 dllkelsnil gfiydvklvp dgkygaqndk gewngmvkel idhradlava pltityvrek 541 vidfskpfmt lgisilyrkp ngtnpgvfsf lnplspdiwm yvllaclgvs cvlfviarft 601 pyewynphpc npdsdvvenn ftllnsfwfg vgalmqqgse lmpkalstri vggiwwfftl 661 iiissytanl aafltverme spidsaddla kqtkieygav rdgstmtffk kskistyekm 721 wafmssrqqt alvrnsdegi qrvlttdyal lmestsieyv tqrncnltqi gglidskgyg 781 vgtpigspyr dkitiailql qeegklhmmk ekwwrgngcp eednkeasal gveniggifi 841 vlaaglvlsv fvaigefiyk srknndieqa fcffyglqck qthptnstsg ttlstdlecg 901 klireergir kqssvhtv';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkiifpilsn pvfrrtvkll lcllwigysq gtthvlrfgg ifeyvesgpm gaeelafrfa 61 vntinrnrtl lpnttltydt qkinlydsfe askkacdqls lgvaaifgps hsssanavqs 121 icnalgvphi qtrwkhqvsd nkdsfyvsly pdfsslsrai ldlvqffkwk tvtvvyddst 181 glirlqelik apsrynlrlk irqlpadtkd akpllkemkr gkefhvifdc shemaagilk 241 qalamgmmte yyhyifttld lfaldvepyr ysgvnmtgfr ilntentqvs siiekwsmer 301 lqappkpdsg lldgfmttda almydavhvv svavqqfpqm tvsslqcnrh kpwrfgtrfm 361 slikeahweg ltgritfnkt nglrtdfdld vislkeegle kigtwdpasg lnmtesqkgk 421 panitdslsn rslivttile epyvlfkksd kplygndrfe gycidllrel stilgftyei 481 rlvedgkyga qddangqwng mvrelidhka dlavaplait yvrekvidfs kpfmtlgisi 541 lyrkpngtnp gvfsflnpls pdiwmyilla ylgvscvlfv iarfspyewy nphpcnpdsd 601 vvennftlln sfwfgvgalm qqgselmpka lstrivggiw wfftliiiss ytanlaaflt 661 vermespids addlakqtki eygavedgat mtffkkskis tydkmwafms srrqsvlvks 721 neegiqrvlt sdyaflmest tiefvtqrnc nltqigglid skgygvgtpm gspyrdkiti 781 ailqlqeegk lhmmkekwwr gngcpeeesk easalgvqni ggifivlaag lvlsvfvavg 841 eflykskkna qlekrsfcsa mveelrmslk cqrrlkhkpq apvivkteev inmhtfndrr 901 lpgketma';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mtapwrrlrs lvweywagll vcafwipdsr gmphvirigg ifeyadgpna qvmnaeehaf 61 rfsaniinrn rtllpnttlt ydiqrihfhd sfeatkkacd qlalgvvaif gpsqgsctna 121 vqsicnalev phiqlrwkhh pldnkdtfyv nlypdyasls haildlvqyl kwrsatvvyd 181 dstglirlqe limapsryni rlkirqlpid sddsrpllke mkrgrefrii fdcshtmaaq 241 ilkqamamgm mteyyhfift tldlyaldle pyrysgvnlt gfrilnvdnp hvsaivekws 301 merlqaaprs esglldgvmm tdaallydav hivsvcyqra pqmtvnslqc hrhkawrfgg 361 rfmnfikeaq wegltgrivf nktsglrtdf dldiislked glekvgvwsp adglniteva 421 kgrgpnvtds ltnrslivtt vleepfvmfr ksdrtlygnd rfegycidll kelahilgfs 481 yeirlvedgk ygaqddkgqw ngmvkelidh kadlavaplt ithvrekaid fskpfmtlgv 541 silyrkpngt npsvfsflnp lspdiwmyvl laylgvscvl fviarfspye wydahpcnpg 601 sevvennftl lnsfwfgmgs lmqqgselmp kalstriigg iwwfftliii ssytanlaaf 661 ltvermespi dsaddlakqt kieygavkdg atmtffkksk istfekmwaf msskpsalvk 721 nneegiqral tadyallmes ttieyvtqrn cnltqiggli dskgygigtp mgspyrdkit 781 iailqlqeed klhimkekww rgsgcpeeen keasalgiqk iggifivlaa glvlsvlvav 841 gefvyklrkt aereqrsfcs tvadeirfsl tcqrrvkhkp qppmmvktda vinmhtfndr 901 rlpgkdsmac stslapvfp';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mprvsaplvl lpawlvmvac sphslriaai lddpmecsrg erlsitlakn rinraperlg 61 kakvevdife llrdseyeta etmcqilpkg vvavlgpsss passsiisni cgekevphfk 121 vapeefvkfq fqrfttlnlh psntdisvav agilnffnct taclicakae cllnlekllr 181 qfliskdtls vrmlddtrdp tpllkeirdd ktatiiihan asmshtillk aaelgmvsay 241 ytyiftnlef slqrmdslvd drvnilgfsi fnqshaffqe faqslnqswq encdhvpftg 301 palssallfd avyavvtavq elnrsqeigv kplscgsaqi wqhgtslmny lrmveleglt 361 ghiefnskgq rsnyalkilq ftrngfrqig qwhvaeglsm dshlyasnis dtlfnttlvv 421 ttilenpylm lkgnhqemeg ndryegfcvd mlkelaeilr fnykirlvgd gvygvpeang 481 twtgmvgeli arkadlavag ltitaerekv idfskpfmtl gisilyrvhm grkpgyfsfl 541 dpfspgvwlf mllaylavsc vlflvarltp yewysphpca qgrcnllvnq yslgnslwfp 601 vggfmqqgst iapralstrc vsgvwwaftl iiissytanl aafltvqrmd vpiesvddla 661 dqtaieygti hggssmtffq nsryqtyqrm wnymyskqps vfvksteegi arvlnsnyaf 721 llestmneyy rqrncnltqi gglldtkgyg igmpvgsvfr defdlailql qennrleilk 781 rkwweggkcp keedhrakgl gmeniggifv vlicglivai fmamleflwt lrhseatevs 841 vcqemvtelr siilcqdsih prrrraavpp prppipeerr prgtatlsng klcgagepdq 901 laqrlaqeaa lvargcthir vcpecrrfqg lrarpspars eeslewektt nssepe';

protein();

} else {

peptidesequence = '1 mpaellllli vafaspscqv lsslrmaail ddqtvcgrge rlalalareq ingiievpak 61 arvevdifel qrdsqyettd tmcqilpkgv vsvlgpsssp asastvshic gekeiphikv 121 gpeetprlqy lrfasvslyp snedvslavs rilksfnyps aslicakaec llrleelvrg 181 flisketlsv rmlddsrdpt pllkeirddk vstiiidana sishlilrka selgmtsafy 241 kyilttmdfp ilhldgived ssnilgfsmf ntshpfypef vrslnmswre nceastylgp 301 alsaalmfda vhvvvsavre lnrsqeigvk plactsaniw phgtslmnyl rmveydgltg 361 rvefnskgqr tnytlrilek srqghreigv wysnrtlamn attldinlsq tlanktlvvt 421 tilenpyvmr rpnfqalsgn erfegfcvdm lrelaellrf ryrlrlvedg lygapepngs 481 wtgmvgelin rkadlavaaf titaerekvi dfskpfmtlg isilyrvhmg rkpgyfsfld 541 pfspavwlfm llaylavscv lflaarlspy ewynphpclr arphilenqy tlgnslwfpv 601 ggfmqqgsei mpralstrcv sgvwwaftli iissytanla afltvqrmev pvesaddlad 661 qtnieygtih agstmtffqn sryqtyqrmw nymqskqpsv fvksteegia rvlnsryafl 721 lestmneyhr rlncnltqig glldtkgygi gmplgspfrd eitlailqlq ennrleilkr 781 kwweggrcpk eedhrakglg meniggifiv licgliiavf vavmefiwst rrsaeseetp 841 alhpaacqcs algprtplke psmllvkvps trvqvafsrt slrqvcpfll qhqlsslywi 901 qatnvqicch fsslkpspdl tfppshrpls sllftalaav gglpdassff fppisscppl 961 qsgigpchst eatlvtsnfh v';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y + 6.187927353e+27;

cursor1z = cursor2z;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

function synapse\_z() {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x;

cursor1z = cursor2z - 6.187927353e+28;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z + 6.187927353e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mehgtllaqp glwtrdtswa llyflcyilp qtapqvlrig gifetvenep vnveelafkf 61 avtsinrnrt lmpnttltyd iqrinlfdsf easrracdql algvaalfgp shsssvsavq 121 sicnalevph iqtrwkhpsv dnkdlfyinl ypdyaaisra ildlvlyynw ktvtvvyeds 181 tglirlqeli kapsryniki kirqlpsgnk dakpllkemk kgkefyvifd cshetaaeil 241 kqilfmgmmt eyyhyffttl dlfaldlely rysgvnmtgf rllnidnphv ssiiekwsme 301 rlqapprpet glldgmmtte aalmydavym vaiashrasq ltvsslqchr hkpwrlgprf 361 mnlikearwd gltghitfnk tnglrkdfdl diislkeegt ekaagevskh lykvwkkigi 421 wnsnsglnmt dsnkdkssni tdslanrtli vttileepyv myrksdkply gndrfegycl 481 dllkelsnil gfiydvklvp dgkygaqndk gewngmvkel idhradlava pltityvrek 541 vidfskpfmt lgisilyrkp ngtnpgvfsf lnplspdiwm yvllaclgvs cvlfviarft 601 pyewynphpc npdsdvvenn ftllnsfwfg vgalmqqgse lmpkalstri vggiwwfftl 661 iiissytanl aafltverme spidsaddla kqtkieygav rdgstmtffk kskistyekm 721 wafmssrqqt alvrnsdegi qrvlttdyal lmestsieyv tqrncnltqi gglidskgyg 781 vgtpigspyr dkitiailql qeegklhmmk ekwwrgngcp eednkeasal gveniggifi 841 vlaaglvlsv fvaigefiyk srknndieqa fcffyglqck qthptnstsg ttlstdlecg 901 klireergir kqssvhtv';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkiifpilsn pvfrrtvkll lcllwigysq gtthvlrfgg ifeyvesgpm gaeelafrfa 61 vntinrnrtl lpnttltydt qkinlydsfe askkacdqls lgvaaifgps hsssanavqs 121 icnalgvphi qtrwkhqvsd nkdsfyvsly pdfsslsrai ldlvqffkwk tvtvvyddst 181 glirlqelik apsrynlrlk irqlpadtkd akpllkemkr gkefhvifdc shemaagilk 241 qalamgmmte yyhyifttld lfaldvepyr ysgvnmtgfr ilntentqvs siiekwsmer 301 lqappkpdsg lldgfmttda almydavhvv svavqqfpqm tvsslqcnrh kpwrfgtrfm 361 slikeahweg ltgritfnkt nglrtdfdld vislkeegle kigtwdpasg lnmtesqkgk 421 panitdslsn rslivttile epyvlfkksd kplygndrfe gycidllrel stilgftyei 481 rlvedgkyga qddangqwng mvrelidhka dlavaplait yvrekvidfs kpfmtlgisi 541 lyrkpngtnp gvfsflnpls pdiwmyilla ylgvscvlfv iarfspyewy nphpcnpdsd 601 vvennftlln sfwfgvgalm qqgselmpka lstrivggiw wfftliiiss ytanlaaflt 661 vermespids addlakqtki eygavedgat mtffkkskis tydkmwafms srrqsvlvks 721 neegiqrvlt sdyaflmest tiefvtqrnc nltqigglid skgygvgtpm gspyrdkiti 781 ailqlqeegk lhmmkekwwr gngcpeeesk easalgvqni ggifivlaag lvlsvfvavg 841 eflykskkna qlekrsfcsa mveelrmslk cqrrlkhkpq apvivkteev inmhtfndrr 901 lpgketma';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mtapwrrlrs lvweywagll vcafwipdsr gmphvirigg ifeyadgpna qvmnaeehaf 61 rfsaniinrn rtllpnttlt ydiqrihfhd sfeatkkacd qlalgvvaif gpsqgsctna 121 vqsicnalev phiqlrwkhh pldnkdtfyv nlypdyasls haildlvqyl kwrsatvvyd 181 dstglirlqe limapsryni rlkirqlpid sddsrpllke mkrgrefrii fdcshtmaaq 241 ilkqamamgm mteyyhfift tldlyaldle pyrysgvnlt gfrilnvdnp hvsaivekws 301 merlqaaprs esglldgvmm tdaallydav hivsvcyqra pqmtvnslqc hrhkawrfgg 361 rfmnfikeaq wegltgrivf nktsglrtdf dldiislked glekvgvwsp adglniteva 421 kgrgpnvtds ltnrslivtt vleepfvmfr ksdrtlygnd rfegycidll kelahilgfs 481 yeirlvedgk ygaqddkgqw ngmvkelidh kadlavaplt ithvrekaid fskpfmtlgv 541 silyrkpngt npsvfsflnp lspdiwmyvl laylgvscvl fviarfspye wydahpcnpg 601 sevvennftl lnsfwfgmgs lmqqgselmp kalstriigg iwwfftliii ssytanlaaf 661 ltvermespi dsaddlakqt kieygavkdg atmtffkksk istfekmwaf msskpsalvk 721 nneegiqral tadyallmes ttieyvtqrn cnltqiggli dskgygigtp mgspyrdkit 781 iailqlqeed klhimkekww rgsgcpeeen keasalgiqk iggifivlaa glvlsvlvav 841 gefvyklrkt aereqrsfcs tvadeirfsl tcqrrvkhkp qppmmvktda vinmhtfndr 901 rlpgkdsmac stslapvfp';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mprvsaplvl lpawlvmvac sphslriaai lddpmecsrg erlsitlakn rinraperlg 61 kakvevdife llrdseyeta etmcqilpkg vvavlgpsss passsiisni cgekevphfk 121 vapeefvkfq fqrfttlnlh psntdisvav agilnffnct taclicakae cllnlekllr 181 qfliskdtls vrmlddtrdp tpllkeirdd ktatiiihan asmshtillk aaelgmvsay 241 ytyiftnlef slqrmdslvd drvnilgfsi fnqshaffqe faqslnqswq encdhvpftg 301 palssallfd avyavvtavq elnrsqeigv kplscgsaqi wqhgtslmny lrmveleglt 361 ghiefnskgq rsnyalkilq ftrngfrqig qwhvaeglsm dshlyasnis dtlfnttlvv 421 ttilenpylm lkgnhqemeg ndryegfcvd mlkelaeilr fnykirlvgd gvygvpeang 481 twtgmvgeli arkadlavag ltitaerekv idfskpfmtl gisilyrvhm grkpgyfsfl 541 dpfspgvwlf mllaylavsc vlflvarltp yewysphpca qgrcnllvnq yslgnslwfp 601 vggfmqqgst iapralstrc vsgvwwaftl iiissytanl aafltvqrmd vpiesvddla 661 dqtaieygti hggssmtffq nsryqtyqrm wnymyskqps vfvksteegi arvlnsnyaf 721 llestmneyy rqrncnltqi gglldtkgyg igmpvgsvfr defdlailql qennrleilk 781 rkwweggkcp keedhrakgl gmeniggifv vlicglivai fmamleflwt lrhseatevs 841 vcqemvtelr siilcqdsih prrrraavpp prppipeerr prgtatlsng klcgagepdq 901 laqrlaqeaa lvargcthir vcpecrrfqg lrarpspars eeslewektt nssepe';

protein();

} else {

peptidesequence = '1 mpaellllli vafaspscqv lsslrmaail ddqtvcgrge rlalalareq ingiievpak 61 arvevdifel qrdsqyettd tmcqilpkgv vsvlgpsssp asastvshic gekeiphikv 121 gpeetprlqy lrfasvslyp snedvslavs rilksfnyps aslicakaec llrleelvrg 181 flisketlsv rmlddsrdpt pllkeirddk vstiiidana sishlilrka selgmtsafy 241 kyilttmdfp ilhldgived ssnilgfsmf ntshpfypef vrslnmswre nceastylgp 301 alsaalmfda vhvvvsavre lnrsqeigvk plactsaniw phgtslmnyl rmveydgltg 361 rvefnskgqr tnytlrilek srqghreigv wysnrtlamn attldinlsq tlanktlvvt 421 tilenpyvmr rpnfqalsgn erfegfcvdm lrelaellrf ryrlrlvedg lygapepngs 481 wtgmvgelin rkadlavaaf titaerekvi dfskpfmtlg isilyrvhmg rkpgyfsfld 541 pfspavwlfm llaylavscv lflaarlspy ewynphpclr arphilenqy tlgnslwfpv 601 ggfmqqgsei mpralstrcv sgvwwaftli iissytanla afltvqrmev pvesaddlad 661 qtnieygtih agstmtffqn sryqtyqrmw nymqskqpsv fvksteegia rvlnsryafl 721 lestmneyhr rlncnltqig glldtkgygi gmplgspfrd eitlailqlq ennrleilkr 781 kwweggrcpk eedhrakglg meniggifiv licgliiavf vavmefiwst rrsaeseetp 841 alhpaacqcs algprtplke psmllvkvps trvqvafsrt slrqvcpfll qhqlsslywi 901 qatnvqicch fsslkpspdl tfppshrpls sllftalaav gglpdassff fppisscppl 961 qsgigpchst eatlvtsnfh v';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

function synapse\_z2() {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x;

cursor1z = cursor2z - 6.187927353e+28;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mehgtllaqp glwtrdtswa llyflcyilp qtapqvlrig gifetvenep vnveelafkf 61 avtsinrnrt lmpnttltyd iqrinlfdsf easrracdql algvaalfgp shsssvsavq 121 sicnalevph iqtrwkhpsv dnkdlfyinl ypdyaaisra ildlvlyynw ktvtvvyeds 181 tglirlqeli kapsryniki kirqlpsgnk dakpllkemk kgkefyvifd cshetaaeil 241 kqilfmgmmt eyyhyffttl dlfaldlely rysgvnmtgf rllnidnphv ssiiekwsme 301 rlqapprpet glldgmmtte aalmydavym vaiashrasq ltvsslqchr hkpwrlgprf 361 mnlikearwd gltghitfnk tnglrkdfdl diislkeegt ekaagevskh lykvwkkigi 421 wnsnsglnmt dsnkdkssni tdslanrtli vttileepyv myrksdkply gndrfegycl 481 dllkelsnil gfiydvklvp dgkygaqndk gewngmvkel idhradlava pltityvrek 541 vidfskpfmt lgisilyrkp ngtnpgvfsf lnplspdiwm yvllaclgvs cvlfviarft 601 pyewynphpc npdsdvvenn ftllnsfwfg vgalmqqgse lmpkalstri vggiwwfftl 661 iiissytanl aafltverme spidsaddla kqtkieygav rdgstmtffk kskistyekm 721 wafmssrqqt alvrnsdegi qrvlttdyal lmestsieyv tqrncnltqi gglidskgyg 781 vgtpigspyr dkitiailql qeegklhmmk ekwwrgngcp eednkeasal gveniggifi 841 vlaaglvlsv fvaigefiyk srknndieqa fcffyglqck qthptnstsg ttlstdlecg 901 klireergir kqssvhtv';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mkiifpilsn pvfrrtvkll lcllwigysq gtthvlrfgg ifeyvesgpm gaeelafrfa 61 vntinrnrtl lpnttltydt qkinlydsfe askkacdqls lgvaaifgps hsssanavqs 121 icnalgvphi qtrwkhqvsd nkdsfyvsly pdfsslsrai ldlvqffkwk tvtvvyddst 181 glirlqelik apsrynlrlk irqlpadtkd akpllkemkr gkefhvifdc shemaagilk 241 qalamgmmte yyhyifttld lfaldvepyr ysgvnmtgfr ilntentqvs siiekwsmer 301 lqappkpdsg lldgfmttda almydavhvv svavqqfpqm tvsslqcnrh kpwrfgtrfm 361 slikeahweg ltgritfnkt nglrtdfdld vislkeegle kigtwdpasg lnmtesqkgk 421 panitdslsn rslivttile epyvlfkksd kplygndrfe gycidllrel stilgftyei 481 rlvedgkyga qddangqwng mvrelidhka dlavaplait yvrekvidfs kpfmtlgisi 541 lyrkpngtnp gvfsflnpls pdiwmyilla ylgvscvlfv iarfspyewy nphpcnpdsd 601 vvennftlln sfwfgvgalm qqgselmpka lstrivggiw wfftliiiss ytanlaaflt 661 vermespids addlakqtki eygavedgat mtffkkskis tydkmwafms srrqsvlvks 721 neegiqrvlt sdyaflmest tiefvtqrnc nltqigglid skgygvgtpm gspyrdkiti 781 ailqlqeegk lhmmkekwwr gngcpeeesk easalgvqni ggifivlaag lvlsvfvavg 841 eflykskkna qlekrsfcsa mveelrmslk cqrrlkhkpq apvivkteev inmhtfndrr 901 lpgketma';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mtapwrrlrs lvweywagll vcafwipdsr gmphvirigg ifeyadgpna qvmnaeehaf 61 rfsaniinrn rtllpnttlt ydiqrihfhd sfeatkkacd qlalgvvaif gpsqgsctna 121 vqsicnalev phiqlrwkhh pldnkdtfyv nlypdyasls haildlvqyl kwrsatvvyd 181 dstglirlqe limapsryni rlkirqlpid sddsrpllke mkrgrefrii fdcshtmaaq 241 ilkqamamgm mteyyhfift tldlyaldle pyrysgvnlt gfrilnvdnp hvsaivekws 301 merlqaaprs esglldgvmm tdaallydav hivsvcyqra pqmtvnslqc hrhkawrfgg 361 rfmnfikeaq wegltgrivf nktsglrtdf dldiislked glekvgvwsp adglniteva 421 kgrgpnvtds ltnrslivtt vleepfvmfr ksdrtlygnd rfegycidll kelahilgfs 481 yeirlvedgk ygaqddkgqw ngmvkelidh kadlavaplt ithvrekaid fskpfmtlgv 541 silyrkpngt npsvfsflnp lspdiwmyvl laylgvscvl fviarfspye wydahpcnpg 601 sevvennftl lnsfwfgmgs lmqqgselmp kalstriigg iwwfftliii ssytanlaaf 661 ltvermespi dsaddlakqt kieygavkdg atmtffkksk istfekmwaf msskpsalvk 721 nneegiqral tadyallmes ttieyvtqrn cnltqiggli dskgygigtp mgspyrdkit 781 iailqlqeed klhimkekww rgsgcpeeen keasalgiqk iggifivlaa glvlsvlvav 841 gefvyklrkt aereqrsfcs tvadeirfsl tcqrrvkhkp qppmmvktda vinmhtfndr 901 rlpgkdsmac stslapvfp';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mprvsaplvl lpawlvmvac sphslriaai lddpmecsrg erlsitlakn rinraperlg 61 kakvevdife llrdseyeta etmcqilpkg vvavlgpsss passsiisni cgekevphfk 121 vapeefvkfq fqrfttlnlh psntdisvav agilnffnct taclicakae cllnlekllr 181 qfliskdtls vrmlddtrdp tpllkeirdd ktatiiihan asmshtillk aaelgmvsay 241 ytyiftnlef slqrmdslvd drvnilgfsi fnqshaffqe faqslnqswq encdhvpftg 301 palssallfd avyavvtavq elnrsqeigv kplscgsaqi wqhgtslmny lrmveleglt 361 ghiefnskgq rsnyalkilq ftrngfrqig qwhvaeglsm dshlyasnis dtlfnttlvv 421 ttilenpylm lkgnhqemeg ndryegfcvd mlkelaeilr fnykirlvgd gvygvpeang 481 twtgmvgeli arkadlavag ltitaerekv idfskpfmtl gisilyrvhm grkpgyfsfl 541 dpfspgvwlf mllaylavsc vlflvarltp yewysphpca qgrcnllvnq yslgnslwfp 601 vggfmqqgst iapralstrc vsgvwwaftl iiissytanl aafltvqrmd vpiesvddla 661 dqtaieygti hggssmtffq nsryqtyqrm wnymyskqps vfvksteegi arvlnsnyaf 721 llestmneyy rqrncnltqi gglldtkgyg igmpvgsvfr defdlailql qennrleilk 781 rkwweggkcp keedhrakgl gmeniggifv vlicglivai fmamleflwt lrhseatevs 841 vcqemvtelr siilcqdsih prrrraavpp prppipeerr prgtatlsng klcgagepdq 901 laqrlaqeaa lvargcthir vcpecrrfqg lrarpspars eeslewektt nssepe';

protein();

} else {

peptidesequence = '1 mpaellllli vafaspscqv lsslrmaail ddqtvcgrge rlalalareq ingiievpak 61 arvevdifel qrdsqyettd tmcqilpkgv vsvlgpsssp asastvshic gekeiphikv 121 gpeetprlqy lrfasvslyp snedvslavs rilksfnyps aslicakaec llrleelvrg 181 flisketlsv rmlddsrdpt pllkeirddk vstiiidana sishlilrka selgmtsafy 241 kyilttmdfp ilhldgived ssnilgfsmf ntshpfypef vrslnmswre nceastylgp 301 alsaalmfda vhvvvsavre lnrsqeigvk plactsaniw phgtslmnyl rmveydgltg 361 rvefnskgqr tnytlrilek srqghreigv wysnrtlamn attldinlsq tlanktlvvt 421 tilenpyvmr rpnfqalsgn erfegfcvdm lrelaellrf ryrlrlvedg lygapepngs 481 wtgmvgelin rkadlavaaf titaerekvi dfskpfmtlg isilyrvhmg rkpgyfsfld 541 pfspavwlfm llaylavscv lflaarlspy ewynphpclr arphilenqy tlgnslwfpv 601 ggfmqqgsei mpralstrcv sgvwwaftli iissytanla afltvqrmev pvesaddlad 661 qtnieygtih agstmtffqn sryqtyqrmw nymqskqpsv fvksteegia rvlnsryafl 721 lestmneyhr rlncnltqig glldtkgygi gmplgspfrd eitlailqlq ennrleilkr 781 kwweggrcpk eedhrakglg meniggifiv licgliiavf vavmefiwst rrsaeseetp 841 alhpaacqcs algprtplke psmllvkvps trvqvafsrt slrqvcpfll qhqlsslywi 901 qatnvqicch fsslkpspdl tfppshrpls sllftalaav gglpdassff fppisscppl 961 qsgigpchst eatlvtsnfh v';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z + 6.187927353e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 3.093963676e+26 \* 1;

density\_parameter\_y = 3.093963676e+26 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

function synaptic\_vesicle\_1() {

lipid1 = 2.475170941e+27;

lipid2 = 2.475170941e+27;

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

lipid\_wall\_x();

lipid\_wall\_y();

lipid\_wall\_z();

cursor2x = cursor2x - 2.475170941e+27;

lipid\_wall\_x();

cursor2x = cursor2x + 2.475170941e+27;

cursor2y = cursor2y - 2.475170941e+27;

lipid\_wall\_y();

cursor2y = cursor2y + 2.475170941e+27;

cursor2z = cursor2z + 2.475170941e+27;

lipid\_wall\_z();

cursor2z = cursor2z - 2.475170941e+27;

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y - 2.475170941e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(2.475170941e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.299464744e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 230);

if (randomizer <= 10) {

peptidesequence = '1 mgelfrseem tlaqlflqse aayccvselg elgkvqfrdl npdvnvfqrk fvnevrrcee 61 mdrklrfvek eirkanipim dtgenpevpf prdmidlean fekienelke intnqealkr 121 nfleltelkf ilrktqqffd emadpdllee sssllepsem grgtplrlgf vagvinreri 181 ptfermlwrv crgnvflrqa eienpledpv tgdyvhksvf iiffqgdqlk nrvkkicegf 241 raslypcpet pqerkemasg vntriddlqm vlnqtedhrq rvlqaaakni rvwfikvrkm 301 kaiyhtlnlc nidvtqkcli aevwcpvtdl dsiqfalrrg tehsgstvps ilnrmqtnqt 361 pptynktnkf tygfqnivda ygigtyrein papytiitfp flfavmfgdf ghgilmtlfa 421 vwmvlresri lsqknenemf stvfsgryii llmgvfsmyt gliyndcfsk slnifgssws 481 vrpmftynwt eetlrgnpvl qlnpalpgvf ggpypfgidp iwniatnklt flnsfkmkms 541 vilgiihmlf gvslslfnhi yfkkplniyf gfipeiifmt slfgylvili fykwtaydah 601 tsenapslli hfinmflfsy pesgysmlys gqkgiqcflv vvallcvpwm llfkplvlrr 661 qylrrkhlgt lnfggirvgn gpteedaeii qhdqlsthse dadepsedev fdfgdtmvhq 721 aihtieyclg cisntasylr lwalslahaq lsevlwtmvi higlsvksla gglvlfffft 781 afatltvail limeglsafl halrlhwvef qnkfysgtgf kflpfsfehi regkfee';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mgslfrsetm claqlflqsg tayeclsalg ekglvqfrdl nqnvssfqrk fvgevkrcee 61 lerilvylvq einradiplp egeasppapp lkqvlemqeq lqklevelre vtknkeklrk 121 nllelieyth mlrvtktfvk rnvefeptye efpslesdsl ldyscmqrlg aklgfvsgli 181 nqgkveafek mlwrvckgyt ivsyaeldes ledpetgevi kwyvflisfw geqighkvkk 241 icdcyhchvy pypntaeerr eiqeglntri qdlytvlhkt edylrqvlck aaesvysrvi 301 qvkkmkaiyh mlnmcsfdvt nkcliaevwc peadlqdlrr aleegsresg atipsfmnii 361 ptketpptri rtnkftegfq nivdaygvgs yrevnpalft iitfpflfav mfgdfghgfv 421 mflfalllvl nenhprlnqs qeimrmffng ryilllmglf svytgliynd cfsksvnlfg 481 sgwnvsamys sshppaehkk mvlwndsvvr hnsilqldps ipgvfrgpyp lgidpiwnla 541 tnrltflnsf kmkmsvilgi ihmtfgvilg ifnhlhfrkk fniylvsipe llfmlcifgy 601 lifmifykwl vfsaetsrva psiliefinm flfpasktsg lytgqeyvqr vllvvtalsv 661 pvlflgkplf llwlhngrsc fgvnrsgytl irkdseeevs llgsqdieeg nhqvedgcre 721 maceefnfge ilmtqvihsi eyclgcisnt asylrlwals lahaqlsdvl wamlmrvglr 781 vdttygvlll lpvialfavl tifillimeg lsaflhairl hwvefqnkfy vgagtkfvpf 841 sfsllsskfn nddsva';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mgsmfrseev alvqlflpta aaytcvsrlg elglvefrdl nasvsafqrr fvvdvrrcee 61 lektftflqe evrraglvlp ppkgrlpapp prdllriqee terlaqelrd vrgnqqalra 121 qlhqlqlhaa vlrqghepql aaahtdgase rtpllqapgg phqdlrvnfv agavephkap 181 alerllwrac rgfliasfre leqplehpvt gepatwmtfl isywgeqigq kirkitdcfh 241 chvfpflqqe earlgalqql qqqsqelqev lgeterflsq vlgrvlqllp pgqvqvhkmk 301 avylalnqcs vstthkclia eawcsvrdlp alqealrdss meegvsavah ripcrdmppt 361 lirtnrftas fqgivdaygv gryqevnpap ytiitfpflf avmfgdvghg llmflfalam 421 vlaenrpavk aaqneiwqtf frgryllllm glfsiytgfi ynecfsrats ifpsgwsvaa 481 manqsgwsda flaqhtmltl dpnvtgvflg pypfgidpiw slaanhlsfl nsfkmkmsvi 541 lgvvhmafgv vlgvfnhvhf gqrhrlllet lpeltfllgl fgylvflviy kwlcvwaara 601 asapsilihf inmflfshsp snrllyprqe vvqatlvvla lamvpilllg tplhllhrhr 661 rrlrrrpadr qeenkaglld lpdasvngws sdeekaggld deeeaelvps evlmhqaiht 721 iefclgcvsn tasylrlwal slahaqlsev lwamvmrigl glgrevgvaa vvlvpifaaf 781 avmtvaillv meglsaflha lrlhwvefqn kfysgtgykl spftfaatdd';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mvsvfrseem clsqlflqve aayccvaelg elglvqfkdl nmnvnsfqrk fvnevrrces 61 lerilrfled emqneivvql lekspltplp remitletvl eklegelqea nqnqqalkqs 121 fleltelkyl lkktqdffet etnladdfft edtsgllelk avpaymtgkl gfiagvinre 181 rmasferllw ricrgnvylk fsemdapled pvtkeeiqkn ifiifyqgeq lrqkikkicd 241 gfratvypcp epaverreml esvnvrledl itvitqtesh rqrllqeaaa nwhswlikvq 301 kmkavyhiln mcnidvtqqc viaeiwfpva datrikrale qgmelsgssm apimttvqsk 361 tapptfnrtn kftagfqniv daygvgsyre inpapytiit fpflfavmfg dcghgtvmll 421 aalwmilner rllsqktdne iwntffhgry lillmgifsi ytgliyndcf skslnifgss 481 wsvqpmfrng twnthvmees lylqldpaip gvyfgnpypf gidpiwnlas nkltflnsyk 541 mkmsvilgiv qmvfgvilsl fnhiyfrrtl niilqfipem ifilclfgyl vfmiifkwcc 601 fdvhvsqhap silihfinmf lfnysdssna plykhqqevq sffvvmalis vpwmllikpf 661 ilrashrksq lqasriqeda teniegdsss pssrsgqrts adthgalddh geefnfgdvf 721 vhqaihtiey clgcisntas ylrlwalsla haqlsevlwt mvmnsglqtr gwggivgvfi 781 ifavfavltv aillimegls aflhalrlhw vefqnkfyvg dgykfspfsf khildgtaee';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 mtglallysg vfvafwacal avgvcytifd lgfrfdvawf ltetspfmws nlgiglaisl 61 svvgaawgiy itgssiiggg vkapriktkn lvsiifceav aiygiimaiv isnmaepfsa 121 tdpkaighrn yhagysmfga gltvglsnlf cgvcvgivgs gaaladaqnp slfvkilive 181 ifgsaiglfg vivailqtsr vkmgd';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 msesksgpey asffavmgas aamvfsalga aygtaksgtg iaamsvmrpe qimksiipvv 61 magiiaiygl vvavliansl nddislyksf lqlgaglsvg lsglaagfai givgdagvrg 121 taqqprlfvg mililifaev lglyglival ilstk';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 msffpelyfn vdngyleglv rglkagvlsq adylnlvqce tledlklhlq stdygnflan 61 easpltvsvi ddrlkekmvv efrhmrnhay eplasfldfi tysymidnvi llitgtlhqr 121 siaelvpkch plgsfeqmea vniaqtpael ynailvdtpl aaffqdcise qdldemniei 181 irntlykayl esfykfctll ggttadamcp ilefeadrra fiitinsfgt elskedrakl 241 fphcgrlype glaqlaradd yeqvknvady ypeykllfeg agsnpgdktl edrffehevk 301 lnklaflnqf hfgvfyafvk lkeqecrniv wiaeciaqrh rakidnyipi f';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mlegaelyfn vdhgyleglv rgckaslltq qdyinlvqce tledlkihlq ttdygnflan 61 htnpltvski dtemrkrlcg efeyfrnhsl eplstfltym tcsymidnvi llmngalqkk 121 svkeilgkch plgrftemea vniaetpsdl fnailietpl apffqdcmse naldelniel 181 lrnklyksyl eafykfcknh gdvtaevmcp ilefeadrra fiitlnsfgt elskedretl 241 yptfgklype glrllaqaed fdqmknvadh ygvykplfea vggsggktle dvfyerevqm 301 nvlafnrqfh ygvfyayvkl keqeirnivw iaecisqrhr tkinsyipil';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 mayhgltvpl ivmsvfwgfv gflvpwfipk gpnrgviitm lvtcsvccyl fwliailaql 61 nplfgpqlkn etiwylkyhw p';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mtahsfalpv iifttfwglv giagpwfvpk gpnrgviitm lvatavccyl fwliailaql 61 nplfgpqlkn etiwyvrflw e';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 mdfsklpkil dedkestfgy vhgvsgpvvt acdmagaamy elvrvghsel vgeiirlegd 61 matiqvyeet sgvsvgdpvl rtgkplsvel gpgimgaifd giqrplsdis sqtqsiyipr 121 gvnvsalsrd ikwdftpckn lrvgshitgg diygivsens likhkimlpp rnrgtvtyia 181 ppgnydtsdv vlelefegvk ekftmvqvwp vrqvrpvtek lpanhplltg qrvldalfpc 241 vqggttaipg afgcgktvis qslskysnsd viiyvgcger gnemsevlrd fpeltmevdg 301 kvesimkrta lvantsnmpv aareasiytg itlseyfrdm gyhvsmmads tsrwaealre 361 isgrlaempa dsgypaylga rlasfyerag rvkclgnper egsvsivgav sppggdfsdp 421 vtsatlgivq vfwgldkkla qrkhfpsvnw lisyskymra ldeyydkhft efvplrtkak 481 eilqeeedla eivqlvgkas laetdkitle vaklikddfl qqngytpydr fcpfyktvgm 541 lsnmiafydm arravettaq sdnkitwsii rehmgdilyk lssmkfkdpl kdgeakiksd 601 yaqlledmqn afrsled';

protein();

} else if (randomizer <= 120) {

peptidesequence = '1 mameidsrpg glpgsscnlg aarehmqavt rnyithprvt yrtvcsvngp lvvldrvkfa 61 qyaeivhftl pdgtqrsgqv levagtkaiv qvfegtsgid arkttceftg dilrtpvsed 121 mlgrvfngsg kpidkgpvvm aedfldingq pinphsriyp eemiqtgisp idvmnsiarg 181 qkipifsaag lphneiaaqi crqaglvkks kavldyhddn faivfaamgv nmetarffks 241 dfeqngtmgn vclflnland ptieriitpr lalttaefla yqcekhvlvi ltdmssyaea 301 lrevsaaree vpgrrgfpgy mytdlatiye ragrvegrgg sitqipiltm pnddithpip 361 dltgfitegq iyvdrqlhnr qiyppinvlp slsrlmksai gegmtrkdhg dvsnqlyacy 421 aigkdvqamk avvgeealts edllyleflq kfeknfinqg pyenrsvfes ldlgwkllri 481 fpkemlkrip qavidefysr egalqdlapd tal';

protein();

} else if (randomizer <= 130) {

peptidesequence = '1 malramrgiv ngaapelpvp tggpavgare qalavsrnyl sqprltyktv sgvngplvil 61 dhvkfpryae ivhltlpdgt krsgqvlevs gskavvqvfe gtsgidakkt sceftgdilr 121 tpvsedmlgr vfngsgkpid rgpvvlaedf ldimgqpinp qcriypeemi qtgisaidgm 181 nsiargqkip ifsaaglphn eiaaqicrqa glvkkskdvv dyseenfaiv faamgvnmet 241 arffksdfee ngsmdnvclf lnlandptie riitprlalt taeflayqce khvlviltdm 301 ssyaealrev saareevpgr rgfpgymytd latiyeragr vegrngsitq ipiltmpndd 361 ithpipdltg yitegqiyvd rqlhnrqiyp pinvlpslsr lmksaigegm trkdhadvsn 421 qlyacyaigk dvqamkavvg eealtsddll yleflqkfer nfiaqgpyen rtvfetldig 481 wqllrifpke mlkripqstl sefyprdsak h';

protein();

} else if (randomizer <= 140) {

peptidesequence = '1 mtefwlisap gektcqqtwe klhaatsknn nlavtskfni pdlkvgtldv lvglsdelak 61 ldafvegvvk kvaqymadvl edskdkvqen llangvdlvt yitrfqwdma kypikqslkn 121 iseiiakgvt qidndlksra saynnlkgnl qnlerknags lltrslaeiv kkddfvldse 181 ylvtllvvvp klnhndwikq yetlaemvvp rssnvlsedq dsylcnvtlf rkavddfrhk 241 arenkfivrd fqyneeemka dkeemnrlst dkkkqfgplv rwlkvnfsea fiawihvkal 301 rvfvesvlry glpvnfqaml lqpnkktlkk lrevlhelyk hldssaaaii dapmdipgln 361 lsqqeyypyv yykidcnlle fk';

protein();

} else if (randomizer <= 150) {

peptidesequence = '1 msefwlisap gdkenlqale rmntvtsksn lsyntkfaip dfkvgtldsl vglsdelgkl 61 dtfaeslirr maqsvvevme dskgkvqehl langvdltsf vthfewdmak ypvkqplvsv 121 vdtiakqlaq iemdlksrta ayntlktnle nlekksmgnl ftrtlsdivs kedfvldsey 181 lvtllvivpk pnysqwqkty eslsdmvvpr stklitedke gglftvtlfr kviedfktka 241 kenkftvref yydekeiere reemarllsd kkqqyqtscv alkkgsstfp dhkvkvtplg 301 npdrpaagqt dreresegeg egpllrwlkv nfseafiawi hikalrvfve svlryglpvn 361 fqavllqphk ksstkrlrev lnsvfrhlde vaatsildas veipglqlnn qdyfpyvyfh 421 idlslld';

protein();

} else if (randomizer <= 160) {

peptidesequence = '1 msgkdrieif psrmaqtimk arlkgaqtgr nllkkksdal tlrfrqilkk iietkmlmge 61 vmreaafsla eakftagdfs ttviqnvnka qvkirakkdn vagvtlpvfe hyhegtdsye 121 ltglarggeq laklkrnyak avellvelas lqtsfvtlde aikitnrrvn aiehviipri 181 ertlayiite lderereefy rlkkiqekkk ilkeksekdl eqrraagevl epanllaeek 241 dedllfe';

protein();

} else if (randomizer <= 170) {

peptidesequence = '1 malsdadvqk qikhmmafie qeanekaeei dakaeeefni ekgrlvqtqr lkimeyyekk 61 ekqieqqkki qmsnlmnqar lkvlrarddl itdllneakq rlskvvkdtt ryqvlldglv 121 lqglyqllep rmivrcrkqd fplvkaavqk aipmykiatk ndvdvqidqe sylpediagg 181 veiyngdrki kvsntlesrl dliaqqmmpe vrgalfgana nrkfld';

protein();

} else if (randomizer <= 180) {

peptidesequence = '1 magrgkliav igdedtvtgf llggigelnk nrhpnflvve kdttineied tfrqflnrdd 61 igiilinqyi aemvrhalda hqqsipavle ipskehpyda akdsilrrar gmftaedlr';

protein();

} else if (randomizer <= 190) {

peptidesequence = '1 masqsqgiqq llqaekraae kvsearkrkn rrlkqakeea qaeieqyrlq rekefkakea 61 aalgsrgscs teveketqek mtilqtyfrq nrdevldnll afvcdirpei henyring';

protein();

} else if (randomizer <= 200) {

protein();

peptidesequence = '1 mtsqsqgihq llqaekrakd kleeakkrkg krlkqakeea mveidqyrmq rdkefrlkqs 61 kimgsqnnls deieeqtlgk iqelnghynk ymesvmnqll smvcdmkpei hvnyratn';

} else if (randomizer <= 210) {

protein();

peptidesequence = '1 mtkmdirgav daavptniia akaaevrank vnwqsylqgq misaedcefi qrfemkrspe 61 ekqemlqteg sqcaktfinl mthickeqtv qyiltmvddm lqenhqrvsi ffdyarcskn 121 tawpyflpml nrqdpftvhm aariiaklaa wgkelmegsd lnyyfnwikt qlssqklrgs 181 gvavetgtvs ssdssqyvqc vagclqlmlr vneyrfawve adgvncimgv lsnkcgfqlq 241 yqmifsiwll afspqmcehl rryniipvls dilqesvkek vtriilaafr nflekstere 301 trqeyalami qckvlkqlen leqqkydded isedikflle klgesvqdls sfdeysselk 361 sgrlewspvh ksekfwrena vrlneknyel lkiltkllev sddpqvlava ahdvgeyvrh 421 yprgkrvieq lggkqlvmnh mhhedqqvry nallavqklm vhnweylgkq lqseqpqtaa 481 ars';

} else if (randomizer <= 220) {

protein();

peptidesequence = '1 mtefwlisap gektcqqtwe klhaatsknn nlavtskfni pdlkvgtldv lvglsdelak 61 ldafvegvvk kvaqymadvl edskdkvqen llangvdlvt yitrfqwdma kypikqslkn 121 iseiiakgvt qidndlksra saynnlkgnl qnlerknags lltrslaeiv kkddfvldse 181 ylvtllvvvp klnhndwikq yetlaemvvp rssnvlsedq dsylcnvtlf rkavddfrhk 241 arenkfivrd fqyneeemka dkeemnrlst dkkkqfgplv rwlkvnfsea fiawihvkal 301 rvfvesvlry glpvnfqaml lqpnkktlkk lrevlhelyk hldssaaaii dapmdipgln 361 lsqqeyypyv yykidcnlle fk';

} else {

protein();

peptidesequence = ' 1 msefwlisap gdkenlqale rmntvtsksn lsyntkfaip dfkvgtldsl vglsdelgkl 61 dtfaeslirr maqsvvevme dskgkvqehl langvdltsf vthfewdmak ypvkqplvsv 121 vdtiakqlaq iemdlksrta ayntlktnle nlekksmgnl ftrtlsdivs kedfvldsey 181 lvtllvivpk pnysqwqkty eslsdmvvpr stklitedke gglftvtlfr kviedfktka 241 kenkftvref yydekeiere reemarllsd kkqqyqtscv alkkgsstfp dhkvkvtplg 301 npdrpaagqt dreresegeg egpllrwlkv nfseafiawi hikalrvfve svlryglpvn 361 fqavllqphk ksstkrlrev lnsvfrhlde vaatsildas veipglqlnn qdyfpyvyfh 421 idlslld';

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 3.093963676e+26 \* 1;

generation\_parameter\_x = Math.round(2.475170941e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(2.475170941e+30 / density\_parameter\_z - 0);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glutamate();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x - 2.475170941e+27;

cursor1z = cursor2z;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(1.299464744e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = '1 mefrqeefrk lagralgklh rllekrqega etlelsadgr pvttqtrdpp vvdctcfglp 61 rryiiaimsg lgfcisfgir cnlgvaivsm vnnstthrgg hvvvqkaqfs wdpetvglih 121 gsffwgyivt qipggficqk faanrvfgfa ivatstlnml ipsaarvhyg cvifvrilqg 181 lvegvtypac hgiwskwapp lersrlatta fcgsyagavv amplagvlvq ysgwssvfyv 241 ygsfgifwyl fwllvsyesp alhpsiseee rkyiedaige saklmnpltk fstpwrrfft 301 smpvyaiiva nfcrswtfyl llisqpayfe evfgfeiskv glvsalphlv mtiivpiggq 361 iadflrsrri msttnvrklm ncggfgmeat lllvvgyshs kgvaisflvl avgfsgfais 421 gfnvnhldia pryasilmgi sngvgtlsgm vcpiivgamt khktreewqy vfliaslvhy 481 ggvifygvfa sgekqpwaep eemseekcgf vghdqlagsd dsemedeaep pgappappps 541 ygathstfqp prppppvrdy';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 masstpsssa tssnagadpn ttnlrpttyd twcgvahgct rklglkicgf lqrtnsleek 61 srlvsafker qssknllsce nsdrdarfrr tetdfsnlfa rdllpaknge eqtvqfllev 121 vdillnyvrk tfdrstkvld fhhphqlleg megfnlelsd hpesleqilv dcrdtlkygv 181 rtghprffnq lstgldiigl agewltstan tnmftyeiap vfvlmeqitl kkmreivgws 241 skdgdgifsp ggaisnmysi maarykyfpe vktkgmaavp klvlftseqs hysikkagaa 301 lgfgtdnvil ikcnergkii padfeakile akqkgyvpfy vnatagttvy gafdpiqeia 361 dicekynlwl hvdaawgggl lmsrkhrhkl ngieransvt wnphkmmgvl lqcsailvke 421 kgilqgcnqm cagylfqpdk qydvsydtgd kaiqcgrhvd ifkfwlmwka kgtvgfenqi 481 nkclelaeyl yakiknreef emvfngepeh tnvcfwyipq slrgvpdspq rreklhkvap 541 kikalmmesg ttmvgyqpqg dkanffrmvi snpaatqsdi dflieeierl gqdl';

protein();

} else {

peptidesequence = '1 magasvkvav rvrpfnsrem srdskciiqm sgstttivnp kqpketpksf sfdysywsht 61 spedinyasq kqvyrdigee mlqhafegyn vcifaygqtg agksytmmgk qekdqqgiip 121 qlcedlfsri ndttndnmsy svevsymeiy cervrdllnp knkgnlrvre hpllgpyved 181 lsklavtsyn diqdlmdsgn kartvaatnm netssrshav fniiftqkrh daetnittek 241 vskislvdla gseradstga kgtrlkegan inkslttlgk visalaemds gpnknkkkkk 301 tdfipyrdsv ltwllrenlg gnsrtamvaa lspadinyde tlstlryadr akqircnavi 361 nedpnnklir elkdevtrlr dllyaqglgd itdmtnalvg mspssslsal ssraasvssl 421 herilfapgs eeaierlket ekiiaelnet weeklrrtea irmerealla emgvamredg 481 gtlgvfspkk tphlvnlned plmsecllyy ikdgitrvgr edgerrqdiv lsghfikeeh 541 cvfrsdsrgg seavvtlepc egadtyvngk kvtepsilrs gnriimgksh vfrfnhpeqa 601 rqerertpca etpaepvdwa faqrellekq gidmkqemeq rlqeledqyr rereeatyll 661 eqqrldyesk lealqkqmds ryypevneee eepedevqwt erecelalwa frkwkwyqft 721 slrdllwgna iflkeanais velkkkvqfq fvlltdtlys plppdllppe aakdretrpf 781 prtivavevq dqkngathyw tleklrqrld lmremydraa evpssviedc dnvvtggdpf 841 ydrfpwfrlv grafvylsnl lypvplvhrv aivsekgevk gflrvavqai sadeeapdyg 901 sgvrqsgtak isfddqhfek fqsescpvvg msrsgtsqee lrivegqgqg advgpsadev 961 nnntcsavpp egllldssek aaldgpldaa ldhlrlgntf tfrvtvlqas sisaeyadif 1021 cqfnfihrhd eafsteplkn tgrgpplgfy hvqniavevt ksfieyiksq pivfevfghy 1081 qqhpfpplck dvlsplrpsr rhfprvmpls kpvpatklst ltrpcpgpch ckydllvyfe 1141 iceleangdy ipavvdhrgg mpcmgtfllh qgiqrritvt llhetgshir wkevrelvvg 1201 rirntpetde slidpnilsl nilssgyihp aqddrtfyqf eaawdssmhn slllnrvtpy 1261 rekiymtlsa yiemenctqp avvtkdfcmv fysrdaklpa srsirnlfgs gslrasesnr 1321 vtgvyelslc hvadagspgm qrrrrrvldt svayvrgeen lagwrprsds lildhqwele 1381 klsllqevek trhylllrek letaqrpvpe alspafseds eshgsssass plsaegrpsp 1441 leapnerqre lavkclrllt htfnreyths hvcvsasesk lsemsvtllr dpsmsplgva 1501 tltpsstcps lvegrygatd lrtpqpcsrp aspepellpe adskklpspa ratetdkepq 1561 rllvpdiqei rvspivskkg ylhflephts gwarrfvvvr rpyaymynsd kdtverfvln 1621 lataqveyse dqqamlktpn tfavctehrg illqaasdkd mhdwlyafnp llagtirskl 1681 srrrsaqmrv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x + 2.475170941e+27 / 2;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(1.299464744e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvseshheal aappvttvat vlpsnatepa spgegkedaf sklkekfmne lhkiplppwa 61 liaiaivavl lvltccfcic kkclfkkknk kkgkekggkn ainmkdvkdl gktmkdqalk 121 dddaetgltd geekeepkee eklgklqysl dydfqnnqll vgiiqaaelp aldmggtsdp 181 yvkvfllpdk kkkfetkvhr ktlnpvfneq ftfkvpysel ggktlvmavy dfdrfskhdi 241 igefkvpmnt vdfghvteew rdlqsaekee qeklgdicfs lryvptagkl tvvileaknl 301 kkmdvgglsd pyvkihlmqn gkrlkkkktt ikkntlnpyy nesfsfevpf eqiqkvqvvv 361 tvldydkigk ndaigkvfvg ynstgaelrh wsdmlanprr piaqwhtlqv eeevdamlav 421 kk';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = cursor2z;

cursor1y = cursor2y;

cursor1x = cursor2x;

}

function synaptic\_vesicle\_2() {

lipid1 = 2.475170941e+27;

lipid2 = 2.475170941e+27;

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

lipid\_wall\_x();

lipid\_wall\_y();

lipid\_wall\_z();

cursor2x = cursor2x - 2.475170941e+27;

lipid\_wall\_x();

cursor2x = cursor2x + 2.475170941e+27;

cursor2y = cursor2y - 2.475170941e+27;

lipid\_wall\_y();

cursor2y = cursor2y + 2.475170941e+27;

cursor2z = cursor2z + 2.475170941e+27;

lipid\_wall\_z();

cursor2z = cursor2z - 2.475170941e+27;

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y - 2.475170941e+27;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(2.475170941e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.299464744e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 230);

if (randomizer <= 10) {

peptidesequence = '1 mgelfrseem tlaqlflqse aayccvselg elgkvqfrdl npdvnvfqrk fvnevrrcee 61 mdrklrfvek eirkanipim dtgenpevpf prdmidlean fekienelke intnqealkr 121 nfleltelkf ilrktqqffd emadpdllee sssllepsem grgtplrlgf vagvinreri 181 ptfermlwrv crgnvflrqa eienpledpv tgdyvhksvf iiffqgdqlk nrvkkicegf 241 raslypcpet pqerkemasg vntriddlqm vlnqtedhrq rvlqaaakni rvwfikvrkm 301 kaiyhtlnlc nidvtqkcli aevwcpvtdl dsiqfalrrg tehsgstvps ilnrmqtnqt 361 pptynktnkf tygfqnivda ygigtyrein papytiitfp flfavmfgdf ghgilmtlfa 421 vwmvlresri lsqknenemf stvfsgryii llmgvfsmyt gliyndcfsk slnifgssws 481 vrpmftynwt eetlrgnpvl qlnpalpgvf ggpypfgidp iwniatnklt flnsfkmkms 541 vilgiihmlf gvslslfnhi yfkkplniyf gfipeiifmt slfgylvili fykwtaydah 601 tsenapslli hfinmflfsy pesgysmlys gqkgiqcflv vvallcvpwm llfkplvlrr 661 qylrrkhlgt lnfggirvgn gpteedaeii qhdqlsthse dadepsedev fdfgdtmvhq 721 aihtieyclg cisntasylr lwalslahaq lsevlwtmvi higlsvksla gglvlfffft 781 afatltvail limeglsafl halrlhwvef qnkfysgtgf kflpfsfehi regkfee';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mgslfrsetm claqlflqsg tayeclsalg ekglvqfrdl nqnvssfqrk fvgevkrcee 61 lerilvylvq einradiplp egeasppapp lkqvlemqeq lqklevelre vtknkeklrk 121 nllelieyth mlrvtktfvk rnvefeptye efpslesdsl ldyscmqrlg aklgfvsgli 181 nqgkveafek mlwrvckgyt ivsyaeldes ledpetgevi kwyvflisfw geqighkvkk 241 icdcyhchvy pypntaeerr eiqeglntri qdlytvlhkt edylrqvlck aaesvysrvi 301 qvkkmkaiyh mlnmcsfdvt nkcliaevwc peadlqdlrr aleegsresg atipsfmnii 361 ptketpptri rtnkftegfq nivdaygvgs yrevnpalft iitfpflfav mfgdfghgfv 421 mflfalllvl nenhprlnqs qeimrmffng ryilllmglf svytgliynd cfsksvnlfg 481 sgwnvsamys sshppaehkk mvlwndsvvr hnsilqldps ipgvfrgpyp lgidpiwnla 541 tnrltflnsf kmkmsvilgi ihmtfgvilg ifnhlhfrkk fniylvsipe llfmlcifgy 601 lifmifykwl vfsaetsrva psiliefinm flfpasktsg lytgqeyvqr vllvvtalsv 661 pvlflgkplf llwlhngrsc fgvnrsgytl irkdseeevs llgsqdieeg nhqvedgcre 721 maceefnfge ilmtqvihsi eyclgcisnt asylrlwals lahaqlsdvl wamlmrvglr 781 vdttygvlll lpvialfavl tifillimeg lsaflhairl hwvefqnkfy vgagtkfvpf 841 sfsllsskfn nddsva';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mgsmfrseev alvqlflpta aaytcvsrlg elglvefrdl nasvsafqrr fvvdvrrcee 61 lektftflqe evrraglvlp ppkgrlpapp prdllriqee terlaqelrd vrgnqqalra 121 qlhqlqlhaa vlrqghepql aaahtdgase rtpllqapgg phqdlrvnfv agavephkap 181 alerllwrac rgfliasfre leqplehpvt gepatwmtfl isywgeqigq kirkitdcfh 241 chvfpflqqe earlgalqql qqqsqelqev lgeterflsq vlgrvlqllp pgqvqvhkmk 301 avylalnqcs vstthkclia eawcsvrdlp alqealrdss meegvsavah ripcrdmppt 361 lirtnrftas fqgivdaygv gryqevnpap ytiitfpflf avmfgdvghg llmflfalam 421 vlaenrpavk aaqneiwqtf frgryllllm glfsiytgfi ynecfsrats ifpsgwsvaa 481 manqsgwsda flaqhtmltl dpnvtgvflg pypfgidpiw slaanhlsfl nsfkmkmsvi 541 lgvvhmafgv vlgvfnhvhf gqrhrlllet lpeltfllgl fgylvflviy kwlcvwaara 601 asapsilihf inmflfshsp snrllyprqe vvqatlvvla lamvpilllg tplhllhrhr 661 rrlrrrpadr qeenkaglld lpdasvngws sdeekaggld deeeaelvps evlmhqaiht 721 iefclgcvsn tasylrlwal slahaqlsev lwamvmrigl glgrevgvaa vvlvpifaaf 781 avmtvaillv meglsaflha lrlhwvefqn kfysgtgykl spftfaatdd';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mvsvfrseem clsqlflqve aayccvaelg elglvqfkdl nmnvnsfqrk fvnevrrces 61 lerilrfled emqneivvql lekspltplp remitletvl eklegelqea nqnqqalkqs 121 fleltelkyl lkktqdffet etnladdfft edtsgllelk avpaymtgkl gfiagvinre 181 rmasferllw ricrgnvylk fsemdapled pvtkeeiqkn ifiifyqgeq lrqkikkicd 241 gfratvypcp epaverreml esvnvrledl itvitqtesh rqrllqeaaa nwhswlikvq 301 kmkavyhiln mcnidvtqqc viaeiwfpva datrikrale qgmelsgssm apimttvqsk 361 tapptfnrtn kftagfqniv daygvgsyre inpapytiit fpflfavmfg dcghgtvmll 421 aalwmilner rllsqktdne iwntffhgry lillmgifsi ytgliyndcf skslnifgss 481 wsvqpmfrng twnthvmees lylqldpaip gvyfgnpypf gidpiwnlas nkltflnsyk 541 mkmsvilgiv qmvfgvilsl fnhiyfrrtl niilqfipem ifilclfgyl vfmiifkwcc 601 fdvhvsqhap silihfinmf lfnysdssna plykhqqevq sffvvmalis vpwmllikpf 661 ilrashrksq lqasriqeda teniegdsss pssrsgqrts adthgalddh geefnfgdvf 721 vhqaihtiey clgcisntas ylrlwalsla haqlsevlwt mvmnsglqtr gwggivgvfi 781 ifavfavltv aillimegls aflhalrlhw vefqnkfyvg dgykfspfsf khildgtaee';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 mtglallysg vfvafwacal avgvcytifd lgfrfdvawf ltetspfmws nlgiglaisl 61 svvgaawgiy itgssiiggg vkapriktkn lvsiifceav aiygiimaiv isnmaepfsa 121 tdpkaighrn yhagysmfga gltvglsnlf cgvcvgivgs gaaladaqnp slfvkilive 181 ifgsaiglfg vivailqtsr vkmgd';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 msesksgpey asffavmgas aamvfsalga aygtaksgtg iaamsvmrpe qimksiipvv 61 magiiaiygl vvavliansl nddislyksf lqlgaglsvg lsglaagfai givgdagvrg 121 taqqprlfvg mililifaev lglyglival ilstk';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 msffpelyfn vdngyleglv rglkagvlsq adylnlvqce tledlklhlq stdygnflan 61 easpltvsvi ddrlkekmvv efrhmrnhay eplasfldfi tysymidnvi llitgtlhqr 121 siaelvpkch plgsfeqmea vniaqtpael ynailvdtpl aaffqdcise qdldemniei 181 irntlykayl esfykfctll ggttadamcp ilefeadrra fiitinsfgt elskedrakl 241 fphcgrlype glaqlaradd yeqvknvady ypeykllfeg agsnpgdktl edrffehevk 301 lnklaflnqf hfgvfyafvk lkeqecrniv wiaeciaqrh rakidnyipi f';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mlegaelyfn vdhgyleglv rgckaslltq qdyinlvqce tledlkihlq ttdygnflan 61 htnpltvski dtemrkrlcg efeyfrnhsl eplstfltym tcsymidnvi llmngalqkk 121 svkeilgkch plgrftemea vniaetpsdl fnailietpl apffqdcmse naldelniel 181 lrnklyksyl eafykfcknh gdvtaevmcp ilefeadrra fiitlnsfgt elskedretl 241 yptfgklype glrllaqaed fdqmknvadh ygvykplfea vggsggktle dvfyerevqm 301 nvlafnrqfh ygvfyayvkl keqeirnivw iaecisqrhr tkinsyipil';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 mayhgltvpl ivmsvfwgfv gflvpwfipk gpnrgviitm lvtcsvccyl fwliailaql 61 nplfgpqlkn etiwylkyhw p';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mtahsfalpv iifttfwglv giagpwfvpk gpnrgviitm lvatavccyl fwliailaql 61 nplfgpqlkn etiwyvrflw e';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 mdfsklpkil dedkestfgy vhgvsgpvvt acdmagaamy elvrvghsel vgeiirlegd 61 matiqvyeet sgvsvgdpvl rtgkplsvel gpgimgaifd giqrplsdis sqtqsiyipr 121 gvnvsalsrd ikwdftpckn lrvgshitgg diygivsens likhkimlpp rnrgtvtyia 181 ppgnydtsdv vlelefegvk ekftmvqvwp vrqvrpvtek lpanhplltg qrvldalfpc 241 vqggttaipg afgcgktvis qslskysnsd viiyvgcger gnemsevlrd fpeltmevdg 301 kvesimkrta lvantsnmpv aareasiytg itlseyfrdm gyhvsmmads tsrwaealre 361 isgrlaempa dsgypaylga rlasfyerag rvkclgnper egsvsivgav sppggdfsdp 421 vtsatlgivq vfwgldkkla qrkhfpsvnw lisyskymra ldeyydkhft efvplrtkak 481 eilqeeedla eivqlvgkas laetdkitle vaklikddfl qqngytpydr fcpfyktvgm 541 lsnmiafydm arravettaq sdnkitwsii rehmgdilyk lssmkfkdpl kdgeakiksd 601 yaqlledmqn afrsled';

protein();

} else if (randomizer <= 120) {

peptidesequence = '1 mameidsrpg glpgsscnlg aarehmqavt rnyithprvt yrtvcsvngp lvvldrvkfa 61 qyaeivhftl pdgtqrsgqv levagtkaiv qvfegtsgid arkttceftg dilrtpvsed 121 mlgrvfngsg kpidkgpvvm aedfldingq pinphsriyp eemiqtgisp idvmnsiarg 181 qkipifsaag lphneiaaqi crqaglvkks kavldyhddn faivfaamgv nmetarffks 241 dfeqngtmgn vclflnland ptieriitpr lalttaefla yqcekhvlvi ltdmssyaea 301 lrevsaaree vpgrrgfpgy mytdlatiye ragrvegrgg sitqipiltm pnddithpip 361 dltgfitegq iyvdrqlhnr qiyppinvlp slsrlmksai gegmtrkdhg dvsnqlyacy 421 aigkdvqamk avvgeealts edllyleflq kfeknfinqg pyenrsvfes ldlgwkllri 481 fpkemlkrip qavidefysr egalqdlapd tal';

protein();

} else if (randomizer <= 130) {

peptidesequence = '1 malramrgiv ngaapelpvp tggpavgare qalavsrnyl sqprltyktv sgvngplvil 61 dhvkfpryae ivhltlpdgt krsgqvlevs gskavvqvfe gtsgidakkt sceftgdilr 121 tpvsedmlgr vfngsgkpid rgpvvlaedf ldimgqpinp qcriypeemi qtgisaidgm 181 nsiargqkip ifsaaglphn eiaaqicrqa glvkkskdvv dyseenfaiv faamgvnmet 241 arffksdfee ngsmdnvclf lnlandptie riitprlalt taeflayqce khvlviltdm 301 ssyaealrev saareevpgr rgfpgymytd latiyeragr vegrngsitq ipiltmpndd 361 ithpipdltg yitegqiyvd rqlhnrqiyp pinvlpslsr lmksaigegm trkdhadvsn 421 qlyacyaigk dvqamkavvg eealtsddll yleflqkfer nfiaqgpyen rtvfetldig 481 wqllrifpke mlkripqstl sefyprdsak h';

protein();

} else if (randomizer <= 140) {

peptidesequence = '1 mtefwlisap gektcqqtwe klhaatsknn nlavtskfni pdlkvgtldv lvglsdelak 61 ldafvegvvk kvaqymadvl edskdkvqen llangvdlvt yitrfqwdma kypikqslkn 121 iseiiakgvt qidndlksra saynnlkgnl qnlerknags lltrslaeiv kkddfvldse 181 ylvtllvvvp klnhndwikq yetlaemvvp rssnvlsedq dsylcnvtlf rkavddfrhk 241 arenkfivrd fqyneeemka dkeemnrlst dkkkqfgplv rwlkvnfsea fiawihvkal 301 rvfvesvlry glpvnfqaml lqpnkktlkk lrevlhelyk hldssaaaii dapmdipgln 361 lsqqeyypyv yykidcnlle fk';

protein();

} else if (randomizer <= 150) {

peptidesequence = '1 msefwlisap gdkenlqale rmntvtsksn lsyntkfaip dfkvgtldsl vglsdelgkl 61 dtfaeslirr maqsvvevme dskgkvqehl langvdltsf vthfewdmak ypvkqplvsv 121 vdtiakqlaq iemdlksrta ayntlktnle nlekksmgnl ftrtlsdivs kedfvldsey 181 lvtllvivpk pnysqwqkty eslsdmvvpr stklitedke gglftvtlfr kviedfktka 241 kenkftvref yydekeiere reemarllsd kkqqyqtscv alkkgsstfp dhkvkvtplg 301 npdrpaagqt dreresegeg egpllrwlkv nfseafiawi hikalrvfve svlryglpvn 361 fqavllqphk ksstkrlrev lnsvfrhlde vaatsildas veipglqlnn qdyfpyvyfh 421 idlslld';

protein();

} else if (randomizer <= 160) {

peptidesequence = '1 msgkdrieif psrmaqtimk arlkgaqtgr nllkkksdal tlrfrqilkk iietkmlmge 61 vmreaafsla eakftagdfs ttviqnvnka qvkirakkdn vagvtlpvfe hyhegtdsye 121 ltglarggeq laklkrnyak avellvelas lqtsfvtlde aikitnrrvn aiehviipri 181 ertlayiite lderereefy rlkkiqekkk ilkeksekdl eqrraagevl epanllaeek 241 dedllfe';

protein();

} else if (randomizer <= 170) {

peptidesequence = '1 malsdadvqk qikhmmafie qeanekaeei dakaeeefni ekgrlvqtqr lkimeyyekk 61 ekqieqqkki qmsnlmnqar lkvlrarddl itdllneakq rlskvvkdtt ryqvlldglv 121 lqglyqllep rmivrcrkqd fplvkaavqk aipmykiatk ndvdvqidqe sylpediagg 181 veiyngdrki kvsntlesrl dliaqqmmpe vrgalfgana nrkfld';

protein();

} else if (randomizer <= 180) {

peptidesequence = '1 magrgkliav igdedtvtgf llggigelnk nrhpnflvve kdttineied tfrqflnrdd 61 igiilinqyi aemvrhalda hqqsipavle ipskehpyda akdsilrrar gmftaedlr';

protein();

} else if (randomizer <= 190) {

peptidesequence = '1 masqsqgiqq llqaekraae kvsearkrkn rrlkqakeea qaeieqyrlq rekefkakea 61 aalgsrgscs teveketqek mtilqtyfrq nrdevldnll afvcdirpei henyring';

protein();

} else if (randomizer <= 200) {

protein();

peptidesequence = '1 mtsqsqgihq llqaekrakd kleeakkrkg krlkqakeea mveidqyrmq rdkefrlkqs 61 kimgsqnnls deieeqtlgk iqelnghynk ymesvmnqll smvcdmkpei hvnyratn';

} else if (randomizer <= 210) {

protein();

peptidesequence = '1 mtkmdirgav daavptniia akaaevrank vnwqsylqgq misaedcefi qrfemkrspe 61 ekqemlqteg sqcaktfinl mthickeqtv qyiltmvddm lqenhqrvsi ffdyarcskn 121 tawpyflpml nrqdpftvhm aariiaklaa wgkelmegsd lnyyfnwikt qlssqklrgs 181 gvavetgtvs ssdssqyvqc vagclqlmlr vneyrfawve adgvncimgv lsnkcgfqlq 241 yqmifsiwll afspqmcehl rryniipvls dilqesvkek vtriilaafr nflekstere 301 trqeyalami qckvlkqlen leqqkydded isedikflle klgesvqdls sfdeysselk 361 sgrlewspvh ksekfwrena vrlneknyel lkiltkllev sddpqvlava ahdvgeyvrh 421 yprgkrvieq lggkqlvmnh mhhedqqvry nallavqklm vhnweylgkq lqseqpqtaa 481 ars';

} else if (randomizer <= 220) {

protein();

peptidesequence = '1 mtefwlisap gektcqqtwe klhaatsknn nlavtskfni pdlkvgtldv lvglsdelak 61 ldafvegvvk kvaqymadvl edskdkvqen llangvdlvt yitrfqwdma kypikqslkn 121 iseiiakgvt qidndlksra saynnlkgnl qnlerknags lltrslaeiv kkddfvldse 181 ylvtllvvvp klnhndwikq yetlaemvvp rssnvlsedq dsylcnvtlf rkavddfrhk 241 arenkfivrd fqyneeemka dkeemnrlst dkkkqfgplv rwlkvnfsea fiawihvkal 301 rvfvesvlry glpvnfqaml lqpnkktlkk lrevlhelyk hldssaaaii dapmdipgln 361 lsqqeyypyv yykidcnlle fk';

} else {

protein();

peptidesequence = ' 1 msefwlisap gdkenlqale rmntvtsksn lsyntkfaip dfkvgtldsl vglsdelgkl 61 dtfaeslirr maqsvvevme dskgkvqehl langvdltsf vthfewdmak ypvkqplvsv 121 vdtiakqlaq iemdlksrta ayntlktnle nlekksmgnl ftrtlsdivs kedfvldsey 181 lvtllvivpk pnysqwqkty eslsdmvvpr stklitedke gglftvtlfr kviedfktka 241 kenkftvref yydekeiere reemarllsd kkqqyqtscv alkkgsstfp dhkvkvtplg 301 npdrpaagqt dreresegeg egpllrwlkv nfseafiawi hikalrvfve svlryglpvn 361 fqavllqphk ksstkrlrev lnsvfrhlde vaatsildas veipglqlnn qdyfpyvyfh 421 idlslld';

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1z = cursor2z;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(2.475170941e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(2.475170941e+30 / density\_parameter\_z - 0);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

acetylcholine();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x - 2.475170941e+27;

cursor1z = cursor2z;

cursor1y = cursor2y;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(1.299464744e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = '1 mefrqeefrk lagralgklh rllekrqega etlelsadgr pvttqtrdpp vvdctcfglp 61 rryiiaimsg lgfcisfgir cnlgvaivsm vnnstthrgg hvvvqkaqfs wdpetvglih 121 gsffwgyivt qipggficqk faanrvfgfa ivatstlnml ipsaarvhyg cvifvrilqg 181 lvegvtypac hgiwskwapp lersrlatta fcgsyagavv amplagvlvq ysgwssvfyv 241 ygsfgifwyl fwllvsyesp alhpsiseee rkyiedaige saklmnpltk fstpwrrfft 301 smpvyaiiva nfcrswtfyl llisqpayfe evfgfeiskv glvsalphlv mtiivpiggq 361 iadflrsrri msttnvrklm ncggfgmeat lllvvgyshs kgvaisflvl avgfsgfais 421 gfnvnhldia pryasilmgi sngvgtlsgm vcpiivgamt khktreewqy vfliaslvhy 481 ggvifygvfa sgekqpwaep eemseekcgf vghdqlagsd dsemedeaep pgappappps 541 ygathstfqp prppppvrdy';

protein();

} else {

peptidesequence = '1 magasvkvav rvrpfnsrem srdskciiqm sgstttivnp kqpketpksf sfdysywsht 61 spedinyasq kqvyrdigee mlqhafegyn vcifaygqtg agksytmmgk qekdqqgiip 121 qlcedlfsri ndttndnmsy svevsymeiy cervrdllnp knkgnlrvre hpllgpyved 181 lsklavtsyn diqdlmdsgn kartvaatnm netssrshav fniiftqkrh daetnittek 241 vskislvdla gseradstga kgtrlkegan inkslttlgk visalaemds gpnknkkkkk 301 tdfipyrdsv ltwllrenlg gnsrtamvaa lspadinyde tlstlryadr akqircnavi 361 nedpnnklir elkdevtrlr dllyaqglgd itdmtnalvg mspssslsal ssraasvssl 421 herilfapgs eeaierlket ekiiaelnet weeklrrtea irmerealla emgvamredg 481 gtlgvfspkk tphlvnlned plmsecllyy ikdgitrvgr edgerrqdiv lsghfikeeh 541 cvfrsdsrgg seavvtlepc egadtyvngk kvtepsilrs gnriimgksh vfrfnhpeqa 601 rqerertpca etpaepvdwa faqrellekq gidmkqemeq rlqeledqyr rereeatyll 661 eqqrldyesk lealqkqmds ryypevneee eepedevqwt erecelalwa frkwkwyqft 721 slrdllwgna iflkeanais velkkkvqfq fvlltdtlys plppdllppe aakdretrpf 781 prtivavevq dqkngathyw tleklrqrld lmremydraa evpssviedc dnvvtggdpf 841 ydrfpwfrlv grafvylsnl lypvplvhrv aivsekgevk gflrvavqai sadeeapdyg 901 sgvrqsgtak isfddqhfek fqsescpvvg msrsgtsqee lrivegqgqg advgpsadev 961 nnntcsavpp egllldssek aaldgpldaa ldhlrlgntf tfrvtvlqas sisaeyadif 1021 cqfnfihrhd eafsteplkn tgrgpplgfy hvqniavevt ksfieyiksq pivfevfghy 1081 qqhpfpplck dvlsplrpsr rhfprvmpls kpvpatklst ltrpcpgpch ckydllvyfe 1141 iceleangdy ipavvdhrgg mpcmgtfllh qgiqrritvt llhetgshir wkevrelvvg 1201 rirntpetde slidpnilsl nilssgyihp aqddrtfyqf eaawdssmhn slllnrvtpy 1261 rekiymtlsa yiemenctqp avvtkdfcmv fysrdaklpa srsirnlfgs gslrasesnr 1321 vtgvyelslc hvadagspgm qrrrrrvldt svayvrgeen lagwrprsds lildhqwele 1381 klsllqevek trhylllrek letaqrpvpe alspafseds eshgsssass plsaegrpsp 1441 leapnerqre lavkclrllt htfnreyths hvcvsasesk lsemsvtllr dpsmsplgva 1501 tltpsstcps lvegrygatd lrtpqpcsrp aspepellpe adskklpspa ratetdkepq 1561 rllvpdiqei rvspivskkg ylhflephts gwarrfvvvr rpyaymynsd kdtverfvln 1621 lataqveyse dqqamlktpn tfavctehrg illqaasdkd mhdwlyafnp llagtirskl 1681 srrrsaqmrv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x + 2.475170941e+27 / 2;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+28 \* 1;

density\_parameter\_y = 6.806720089e+28 \* 1;

density\_parameter\_z = 6.806720089e+28;

generation\_parameter\_x = Math.round(1.299464744e+30 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvseshheal aappvttvat vlpsnatepa spgegkedaf sklkekfmne lhkiplppwa 61 liaiaivavl lvltccfcic kkclfkkknk kkgkekggkn ainmkdvkdl gktmkdqalk 121 dddaetgltd geekeepkee eklgklqysl dydfqnnqll vgiiqaaelp aldmggtsdp 181 yvkvfllpdk kkkfetkvhr ktlnpvfneq ftfkvpysel ggktlvmavy dfdrfskhdi 241 igefkvpmnt vdfghvteew rdlqsaekee qeklgdicfs lryvptagkl tvvileaknl 301 kkmdvgglsd pyvkihlmqn gkrlkkkktt ikkntlnpyy nesfsfevpf eqiqkvqvvv 361 tvldydkigk ndaigkvfvg ynstgaelrh wsdmlanprr piaqwhtlqv eeevdamlav 421 kk';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = cursor2z;

cursor1y = cursor2y;

cursor1x = cursor2x;

}

function lipid\_wall\_x() {

if (unused\_variable == 5) {

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

generation\_parameter\_3 = Math.round(lipid1 / density\_parameter\_y - 0);

while (cursor1z < cursor2z + lipid2) {

for (countl1 = 0; countl1 < generation\_parameter\_3; countl1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = cursor2x + 1.546981838e+26;

cursor1y = cursor2y;

cursor1z = cursor2z;

generation\_parameter\_3 = Math.round(lipid1 / density\_parameter\_y - 0);

while (cursor1z < cursor2z + lipid2) {

for (countl1 = 0; countl1 < generation\_parameter\_3; countl1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function lipid\_wall\_y() {

if (unused\_variable == 5) {

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

generation\_parameter\_x = Math.round(lipid1 / density\_parameter\_x - 0);

for (countl1 = 0; countl1 < generation\_parameter\_x; countl1++) {

while (cursor1z < cursor2z + (lipid2 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = cursor2x;

cursor1y = cursor2y + 1.546981838e+26 \* 1;

cursor1z = cursor2z;

generation\_parameter\_x = Math.round(lipid1 / density\_parameter\_x - 0);

for (countl1 = 0; countl1 < generation\_parameter\_x; countl1++) {

while (cursor1z < cursor2z + (lipid2 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

}

function lipid\_wall\_z() {

if (unused\_variable == 5) {

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

generation\_parameter\_x = Math.round(lipid1 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(lipid2 / density\_parameter\_y - 0);

for (countl1 = 0; countl1 < generation\_parameter\_x; countl1++) {

for (countl2 = 0; countl2 < generation\_parameter\_3; countl2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z + 1.546981838e+26;

generation\_parameter\_x = Math.round(lipid1 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(lipid2 / density\_parameter\_y - 0);

for (countl1 = 0; countl1 < generation\_parameter\_x; countl1++) {

for (countl2 = 0; countl2 < generation\_parameter\_3; countl2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

}

function vertebrate\_brain() {

brainx = centerpoint\_x - (sizemid \* 0.5 - 3.527118591e+32 / 2);

brainy = centerpoint\_y - (sizemin \* 0.77 - 2.475170941e+32);

brainz = centerpoint\_z + sizemax \* 0.771;

default\_brain();

}

function human\_brain() {

brainx = centerpoint\_x - (sizemid \* 0.5 - 3.527118591e+32 / 2);

brainy = centerpoint\_y - sizemin \* 0.02;

brainz = centerpoint\_z + sizemax \* 0.941;

default\_brain();

}

function fly\_brain() {

brainx = centerpoint\_x - (sizemid \* 0.5 - 8.663098295e+30);

brainy = centerpoint\_y - sizemin \* 0.4;

brainz = centerpoint\_z + sizemax \* 0.1;

cursor2x = brainx;

cursor2y = brainy;

cursor2z = brainz;

lipid1 = 9.900683765e+30;

lipid2 = 7.425512824e+30;

lipid\_wall\_x();

cursor2x = brainx - 1.732619659e+31;

cursor2y = brainy;

cursor2z = brainz;

lipid\_wall\_x();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.085906752e+25 \* 1;

cedey = 1.546981838e+26 \* 1;

cedez = 4.085906752e+25;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

fly\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.546981838e+26, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.546981838e+26, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 4.085906752e+25];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 4.085906752e+25];

fly\_brain\_in();

loc1 = [cursor1z + 4.085906752e+25, cursor1y, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z - 4.085906752e+25, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true && brain2 == true) {

randomize\_lipid\_1();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.085906752e+25 \* 1;

cedey = 1.546981838e+26 \* 1;

cedez = 4.085906752e+25;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

fly\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.546981838e+26, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.546981838e+26, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 4.085906752e+25];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 4.085906752e+25];

fly\_brain\_in();

loc1 = [cursor1z + 4.085906752e+25, cursor1y, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z - 4.085906752e+25, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true && brain2 == true) {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x + 1.546981838e+26;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor2y;

if (unused\_variable == 5) {

cursor1x = cursor2x - (1.546981838e+26 + 4.085906752e+25);

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor2y;

}

if (unused\_variable == 5) {

cursor1z = cursor2z + (1.546981838e+26 + 4.085906752e+25);

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor2y;

}

if (unused\_variable == 5) {

cursor1z = cursor2z - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor2y;

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.712756412e+29 \* 1;

cedey = 3.712756412e+29 \* 1;

cedez = 3.712756412e+29;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 1.856378206e+29, cursor1y - 1.856378206e+29, cursor1x - 1.856378206e+29];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 1.856378206e+29, cursor1y - 1.856378206e+29, cursor1x - 1.856378206e+29];

obtain\_fly\_neuron();

if (neuron1 == neuron2) {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1y = cursor1y - 6.187927353e+26;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 3.279601497e+27;

generation\_parameter\_x = Math.round(3000);

generation\_parameter\_3 = Math.round(3);

generation\_parameter\_z = Math.round(6);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mparletcis dldcasssgs dlsgfltdee dcarlqqaas asgppaparr gapnisrase 61 vpgaqddeqe rrrrrgrtrv rseallhslr rsrrvkandr ernrmhnlna aldalrsvlp 121 sfpddtkltk ietlrfayny iwalaetlrl adqglpggga rerllppqcv pclpgppspa 181 sdaeswgsga aaasplsdps spaasedfty rpgdpvfsfp slpkdllhtt pcfipyh';

protein();

} else if (randomizer <= 40) {

peptidesequence = 'MFVKSETLELKEEEDVLVLLGSASPALAALTPLSSSADEEEEEE PGASGGARRQRGAEAGQGARGGVAAGAEGCRPARLLGLVHDCKRRPSRARAVSRGAKT AETVQRIKKTRRLKANNRERNRMHNLNAALDALREVLPTFPEDAKLTKIETLRFAHNY IWALTETLRLADHCGGGGGGLPGALFSEAVLLSPGGASAALSSSGDSPSPASTWSCTN SPAPSSSVSSNSTSPYSCTLSPASPAGSDMDYWQPPPPDKHRYAPHLPIARDCI';

protein();

} else if (randomizer <= 60) {

peptidesequence = 'MTPQPSGAPTVQVTRETERSFPRASEDEVTCPTSAPPSPTRTRG NCAEAEEGGCRGAPRKLRARRGGRSRPKSELALSKQRRSRRKKANDRERNRMHNLNSA LDALRGVLPTFPDDAKLTKIETLRFAHNYIWALTQTLRIADHSLYALEPPAPHCGELG SPGGSPGDWGSLYSPVSQAGSLSPAASLEERPGLLGATFSACLSPGSLAFSDFL';

protein();

} else if (randomizer <= 80) {

peptidesequence = 'MTKSYSESGLMGEPQPQGPPSWTDECLSSQDEEHEADKKEDDLE TMNAEEDSLRNGGEEEDEDEDLEEEEEEEEEDDDQKPKRRGPKKKKMTKARLERFKLR RMKANARERNRMHGLNAALDNLRKVVPCYSKTQKLSKIETLRLAKNYIWALSEILRSG KSPDLVSFVQTLCKGLSQPTTNLVAGCLQLNPRTFLPEQNQDMPPHLPTASASFPVHP YSYQSPGLPSPPYGTMDSSHVFHVKPPPHAYSAALEPFFESPLTDCTSPSFDGPLSPP LSINGNFSFKHEPSAEFEKNYAFTMHYPAATLAGAQSHGSIFSGTAAPRCEIPIDNIM SFDSHSHHERVMSAQLNAIFHD';

protein();

} else {

peptidesequence = 'MLTRLFSEPGLLSDVPKFASWGDGEDDEPRSDKGDAPPPPPPAP GPGAPGPARAAKPVPLRGEEGTEATLAEVKEEGELGGEEEEEEEEEEGLDEAEGERPK KRGPKKRKMTKARLERSKLRRQKANARERNRMHDLNAALDNLRKVVPCYSKTQKLSKI ETLRLAKNYIWALSEILRSGKRPDLVSYVQTLCKGLSQPTTNLVAGCLQLNSRNFLTE QGADGAGRFHGSGGPFAMHPYPYPCSRLAGAQCQAAGGLGGGAAHALRTHGYCAAYET LYAAAGGGGASPDYNSSEYEGPLSPPLCLNGNFSLKQDSSPDHEKSYHYSMHYSALPG SRPTGHGLVFGSSAVRGGVHSENLLSYDMHLHHDRGPMYEELNAFFHN';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - density\_parameter\_z \* generation\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

nucleoid\_size\_y = 1.856378206e+29;

nucleoid\_size\_z = 1.856378206e+29;

nucleoid();

cursor1x = cursor2x - 1.23758547e+29;

cursor1y = cursor2y - 1.862566133e+29;

cursor1z = cursor2z;

vital\_components();

cursor1x = cursor2x - 1.23758547e+29;

cursor1y = cursor2y - 6.187927353e+28;

cursor1z = cursor2z;

vital\_components2();

cursor1x = cursor2x;

cursor1y = cursor2y - 6.187927353e+28;

cursor1z = cursor2z;

vital\_components3();

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 6.187927353e+27, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 6.187927353e+27, cursor1y, cursor1x];

obtain\_fly\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_z();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z, cursor1y - 6.187927353e+27, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z, cursor1y - 6.187927353e+27, cursor1x];

obtain\_fly\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_y();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+27];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+27];

obtain\_fly\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_x();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 6.187927353e+29 \* 1;

cedey = 1.23758547e+29 \* 1;

cedez = 1.856378206e+29;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 6.187927353e+28, cursor1y - 4.331549147e+28, cursor1x - 4.331549147e+28];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 6.187927353e+28, cursor1y - 4.331549147e+28, cursor1x - 4.331549147e+28];

obtain\_fly\_neuron();

if (neuron1 == neuron2) {

mitochondrion();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.950341882e+27 \* 1;

cedey = 4.950341882e+27 \* 1;

cedez = 4.950341882e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 4.950341882e+27, cursor1y - 4.950341882e+27, cursor1x - 4.950341882e+27];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_fly\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 4.950341882e+27, cursor1y - 4.950341882e+27, cursor1x - 4.950341882e+27];

obtain\_fly\_neuron();

if (neuron1 == neuron2) {

synaptic\_vesicle\_1();

}

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 1.05194765e+27 \* 1;

cedey = 1.05194765e+27 \* 1;

cedez = 1.05194765e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 1.113826923e+27, cursor1y, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.113826923e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.113826923e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 1.113826923e+27];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 1.113826923e+27];

fly\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.331549147e+26 \* 1;

cedey = 4.331549147e+26 \* 1;

cedez = 4.331549147e+26;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.803131091e+27 \* 1;

cedey = 2.803131091e+27 \* 1;

cedez = 2.803131091e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 3.093963676e+27, cursor1y, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y - 3.093963676e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y + 3.093963676e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 3.093963676e+27];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 3.093963676e+27];

fly\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.666996689e+27 \* 1;

cedey = 2.666996689e+27 \* 1;

cedez = 2.666996689e+27;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 3.093963676e+27, cursor1y, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y - 3.093963676e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y + 3.093963676e+27, cursor1x];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 3.093963676e+27];

fly\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 3.093963676e+27];

fly\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.537050215e+26 \* 1;

cedey = 2.537050215e+26 \* 1;

cedez = 2.537050215e+26;

repeat\_x = Math.round(1.732619659e+31 / cedex - 0);

repeat\_y = Math.round(9.900683765e+30 / cedey - 0);

repeat\_z = Math.round(7.425512824e+30 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

fly\_brain\_in();

if (brain1 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

heme();

} else {

peptidesequence = '1 merpepelir qswravsrsp lehgtvlfar lfalepdllp lfqyncrqfs spedclsspe 61 fldhirkvml vidaavtnve dlssleeyla slgrkhravg vklssfstvg esllymlekc 121 lgpaftpatr aawsqlygav vqamsrgwdg e';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

}

function default\_brain() {

if (unused\_variable == 5) {

spawnerx = brainx;

spawnery = brainy - 2.475170942e+32;

spawnerz = brainz + 3.520930664e+32;

repeat\_endcg1 = Math.round(3.527118591e+32 / 6.187927353e+29);

for (countcg1 = 0; countcg1 < repeat\_endcg1; countcg1++) {

spawnerx = spawnerx - 6.187927353e+29;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

neuroblast();

} else {

glial\_cell\_1();

}

}

}

if (unused\_variable == 5) {

spawnerx = brainx;

spawnery = brainy - 2.475170942e+32;

spawnerz = brainz + 1.825438569e+32;

repeat\_endcg1 = Math.round(3.527118591e+32 / 6.187927353e+29);

for (countcg1 = 0; countcg1 < repeat\_endcg1; countcg1++) {

spawnerx = spawnerx - 6.187927353e+29;

randomizer = math\_random\_int(1, 100);

pineocyte();

}

}

if (unused\_variable == 5) {

spawnerx = brainx;

spawnery = brainy - 2.475170942e+32;

spawnerz = brainz + (3.520930664e+32 + 1.23758547e+30);

repeat\_endcg1 = Math.round(3.527118591e+32 / 6.187927353e+29);

for (countcg1 = 0; countcg1 < repeat\_endcg1; countcg1++) {

spawnerx = spawnerx - 6.187927353e+29;

randomizer = math\_random\_int(1, 100);

hypothalamocyte();

}

}

if (unused\_variable == 5) {

spawnerx = brainx;

spawnery = brainy - 2.475170942e+32;

cursor1z = brainz + (3.520930664e+32 + (1.23758547e+30 + 1.23758547e+30));

repeat\_endcg1 = Math.round(3.527118591e+32 / 6.187927353e+29);

for (countcg1 = 0; countcg1 < repeat\_endcg1; countcg1++) {

spawnerx = spawnerx - 6.187927353e+29;

randomizer = math\_random\_int(1, 100);

pitocyte();

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.085906752e+25 \* 1;

cedey = 1.546981838e+26 \* 1;

cedez = 4.085906752e+25;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.546981838e+26, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.546981838e+26, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 4.085906752e+25];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 4.085906752e+25];

mouse\_brain\_in();

loc1 = [cursor1z + 4.085906752e+25, cursor1y, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z - 4.085906752e+25, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true && brain2 == true) {

randomize\_lipid\_1();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.085906752e+25 \* 1;

cedey = 1.546981838e+26 \* 1;

cedez = 4.085906752e+25;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.546981838e+26, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.546981838e+26, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 4.085906752e+25];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 4.085906752e+25];

mouse\_brain\_in();

loc1 = [cursor1z + 4.085906752e+25, cursor1y, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z - 4.085906752e+25, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true && brain2 == true) {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1x = cursor2x + 1.546981838e+26;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_5();

cursor1y = cursor2y;

if (unused\_variable == 5) {

cursor1x = cursor2x - (1.546981838e+26 + 4.085906752e+25);

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_6();

cursor1y = cursor2y;

}

if (unused\_variable == 5) {

cursor1z = cursor2z + (1.546981838e+26 + 4.085906752e+25);

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor1y - 4.085906752e+25;

randomize\_lipid\_4();

cursor1y = cursor2y;

}

if (unused\_variable == 5) {

cursor1z = cursor2z - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor1y - 1.546981838e+26;

randomize\_lipid\_3();

cursor1y = cursor2y;

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.712756412e+29 \* 1;

cedey = 3.712756412e+29 \* 1;

cedez = 3.712756412e+29;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 1.856378206e+29, cursor1y - 1.856378206e+29, cursor1x - 1.856378206e+29];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 1.856378206e+29, cursor1y - 1.856378206e+29, cursor1x - 1.856378206e+29];

obtain\_mouse\_neuron();

if (neuron1 == neuron2) {

cursor2x = cursor1x;

cursor2y = cursor1y;

cursor2z = cursor1z;

cursor1y = cursor1y - 6.187927353e+26;

if (unused\_variable == 5) {

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 3.279601497e+27;

generation\_parameter\_x = Math.round(3000);

generation\_parameter\_3 = Math.round(3);

generation\_parameter\_z = Math.round(6);

for (countb1 = 0; countb1 < generation\_parameter\_3; countb1++) {

for (countb2 = 0; countb2 < generation\_parameter\_x; countb2++) {

for (countb3 = 0; countb3 < generation\_parameter\_z; countb3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mparletcis dldcasssgs dlsgfltdee dcarlqqaas asgppaparr gapnisrase 61 vpgaqddeqe rrrrrgrtrv rseallhslr rsrrvkandr ernrmhnlna aldalrsvlp 121 sfpddtkltk ietlrfayny iwalaetlrl adqglpggga rerllppqcv pclpgppspa 181 sdaeswgsga aaasplsdps spaasedfty rpgdpvfsfp slpkdllhtt pcfipyh';

protein();

} else if (randomizer <= 40) {

peptidesequence = 'MFVKSETLELKEEEDVLVLLGSASPALAALTPLSSSADEEEEEE PGASGGARRQRGAEAGQGARGGVAAGAEGCRPARLLGLVHDCKRRPSRARAVSRGAKT AETVQRIKKTRRLKANNRERNRMHNLNAALDALREVLPTFPEDAKLTKIETLRFAHNY IWALTETLRLADHCGGGGGGLPGALFSEAVLLSPGGASAALSSSGDSPSPASTWSCTN SPAPSSSVSSNSTSPYSCTLSPASPAGSDMDYWQPPPPDKHRYAPHLPIARDCI';

protein();

} else if (randomizer <= 60) {

peptidesequence = 'MTPQPSGAPTVQVTRETERSFPRASEDEVTCPTSAPPSPTRTRG NCAEAEEGGCRGAPRKLRARRGGRSRPKSELALSKQRRSRRKKANDRERNRMHNLNSA LDALRGVLPTFPDDAKLTKIETLRFAHNYIWALTQTLRIADHSLYALEPPAPHCGELG SPGGSPGDWGSLYSPVSQAGSLSPAASLEERPGLLGATFSACLSPGSLAFSDFL';

protein();

} else if (randomizer <= 80) {

peptidesequence = 'MTKSYSESGLMGEPQPQGPPSWTDECLSSQDEEHEADKKEDDLE TMNAEEDSLRNGGEEEDEDEDLEEEEEEEEEDDDQKPKRRGPKKKKMTKARLERFKLR RMKANARERNRMHGLNAALDNLRKVVPCYSKTQKLSKIETLRLAKNYIWALSEILRSG KSPDLVSFVQTLCKGLSQPTTNLVAGCLQLNPRTFLPEQNQDMPPHLPTASASFPVHP YSYQSPGLPSPPYGTMDSSHVFHVKPPPHAYSAALEPFFESPLTDCTSPSFDGPLSPP LSINGNFSFKHEPSAEFEKNYAFTMHYPAATLAGAQSHGSIFSGTAAPRCEIPIDNIM SFDSHSHHERVMSAQLNAIFHD';

protein();

} else {

peptidesequence = 'MLTRLFSEPGLLSDVPKFASWGDGEDDEPRSDKGDAPPPPPPAP GPGAPGPARAAKPVPLRGEEGTEATLAEVKEEGELGGEEEEEEEEEEGLDEAEGERPK KRGPKKRKMTKARLERSKLRRQKANARERNRMHDLNAALDNLRKVVPCYSKTQKLSKI ETLRLAKNYIWALSEILRSGKRPDLVSYVQTLCKGLSQPTTNLVAGCLQLNSRNFLTE QGADGAGRFHGSGGPFAMHPYPYPCSRLAGAQCQAAGGLGGGAAHALRTHGYCAAYET LYAAAGGGGASPDYNSSEYEGPLSPPLCLNGNFSLKQDSSPDHEKSYHYSMHYSALPG SRPTGHGLVFGSSAVRGGVHSENLLSYDMHLHHDRGPMYEELNAFFHN';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

nucleoid\_size\_y = 1.856378206e+29;

nucleoid\_size\_z = 1.856378206e+29;

nucleoid();

cursor1x = cursor2x - 1.23758547e+29;

cursor1y = cursor2y - 1.862566133e+29;

cursor1z = cursor2z;

vital\_components();

cursor1x = cursor2x - 1.23758547e+29;

cursor1y = cursor2y - 6.187927353e+28;

cursor1z = cursor2z;

vital\_components2();

cursor1x = cursor2x;

cursor1y = cursor2y - 6.187927353e+28;

cursor1z = cursor2z;

vital\_components3();

cursor1x = cursor2x;

cursor1y = cursor2y;

cursor1z = cursor2z;

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 6.187927353e+27, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 6.187927353e+27, cursor1y, cursor1x];

obtain\_mouse\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_z2();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z, cursor1y - 6.187927353e+27, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z, cursor1y - 6.187927353e+27, cursor1x];

obtain\_mouse\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_y2();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 3.162030877e+27 \* 1;

cedey = 3.162030877e+27 \* 1;

cedez = 3.162030877e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+27];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z, cursor1y, cursor1x - 6.187927353e+27];

obtain\_mouse\_neuron();

if (neuron1 != neuron2) {

set\_earth\_rotation();

synapse\_x();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 6.187927353e+29 \* 1;

cedey = 1.23758547e+29 \* 1;

cedez = 1.856378206e+29;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 6.187927353e+28, cursor1y - 4.331549147e+28, cursor1x - 4.331549147e+28];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 6.187927353e+28, cursor1y - 4.331549147e+28, cursor1x - 4.331549147e+28];

obtain\_mouse\_neuron();

if (neuron1 == neuron2) {

mitochondrion();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.950341882e+27 \* 1;

cedey = 4.950341882e+27 \* 1;

cedez = 4.950341882e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z - 4.085906751e+25, cursor1y - 1.546981837e+26, cursor1x + 4.085906751e+25];

exempt = [null];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

brain1 = false;

loc1 = [cursor1z + 4.950341882e+27, cursor1y - 4.950341882e+27, cursor1x - 4.950341882e+27];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z, cursor1y, cursor1x];

obtain\_mouse\_neuron();

neuron2 = neuron1;

loc1 = [cursor1z + 4.950341882e+27, cursor1y - 4.950341882e+27, cursor1x - 4.950341882e+27];

obtain\_mouse\_neuron();

if (neuron1 == neuron2) {

synaptic\_vesicle\_1();

}

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 1.05194765e+27 \* 1;

cedey = 1.05194765e+27 \* 1;

cedez = 1.05194765e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 1.113826923e+27, cursor1y, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y - 1.113826923e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y + 1.113826923e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 1.113826923e+27];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 1.113826923e+27];

mouse\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 4.331549147e+26 \* 1;

cedey = 4.331549147e+26 \* 1;

cedez = 4.331549147e+26;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.803131091e+27 \* 1;

cedey = 2.803131091e+27 \* 1;

cedez = 2.803131091e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 3.093963676e+27, cursor1y, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y - 3.093963676e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y + 3.093963676e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 3.093963676e+27];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 3.093963676e+27];

mouse\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.666996689e+27 \* 1;

cedey = 2.666996689e+27 \* 1;

cedez = 2.666996689e+27;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

loc1 = [cursor1z - 3.093963676e+27, cursor1y, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y - 3.093963676e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y + 3.093963676e+27, cursor1x];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x + 3.093963676e+27];

mouse\_brain\_in();

loc1 = [cursor1z, cursor1y, cursor1x - 3.093963676e+27];

mouse\_brain\_in();

if (brain2 == true) {

randomizer = math\_random\_int(1, 100);

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = brainx;

cursor1y = brainy;

cursor1z = brainz;

}

cedex = 2.537050215e+26 \* 1;

cedey = 2.537050215e+26 \* 1;

cedez = 2.537050215e+26;

repeat\_x = Math.round(3.527118591e+32 / cedex - 0);

repeat\_y = Math.round(2.475170941e+32 / cedey - 0);

repeat\_z = Math.round(3.520930664e+32 / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_y; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_x; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

brain1 = false;

brain2 = false;

loc1 = [cursor1z, cursor1y, cursor1x];

mouse\_brain\_in();

if (brain1 == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

heme();

} else {

peptidesequence = '1 merpepelir qswravsrsp lehgtvlfar lfalepdllp lfqyncrqfs spedclsspe 61 fldhirkvml vidaavtnve dlssleeyla slgrkhravg vklssfstvg esllymlekc 121 lgpaftpatr aawsqlygav vqamsrgwdg e';

protein();

}

}

}

cursor1z = cursor1z + cedez;

}

cursor1z = cursor1z - repeat\_z \* cedez;

cursor1x = cursor1x - cedex;

}

cursor1x = cursor1x + cedex \* repeat\_x;

cursor1y = cursor1y - cedey;

}

}

}

function brain\_junction\_1() {

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 76640 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 131961 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 76640 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 131961 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 76074 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 129031 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 76074 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 129031 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 79352 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 132227 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 79352 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 132227 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 78630 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 137112 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 78630 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 137112 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 81429 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 133365 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 81429 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 133365 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 81533 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 137594 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 81533 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 137594 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 79247 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 129109 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 79247 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 129109 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 82272 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 129335 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 82272 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 129335 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 80308 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 123072 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 80308 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 123072 \* 1.23758547e+26)) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (7.425512824e+30 - 6.187927353e+26), brainy - (100000 \* 1.23758547e+26 - 85115 \* 1.23758547e+26), brainx - (60000 \* 1.23758547e+26 + 124115 \* 1.23758547e+26), brainz + (7.425512824e+30 + 6.187927353e+26), (brainy - (100000 \* 1.23758547e+26 - 85115 \* 1.23758547e+26)) - 1.23758547e+29, (brainx - (60000 \* 1.23758547e+26 + 124115 \* 1.23758547e+26)) - 1.23758547e+29]);

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - (0 \* 1 - 0);

cursor1z = brainz + (7.425512824e+30 - 1.546981839e+26);

generation\_parameter\_x = Math.round(1.732619659e+31 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(9.900683765e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

fly\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_3();

}

} else {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_3();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - 0 \* 1;

cursor1z = brainz + (7.425512824e+30 \* 1 - 0);

generation\_parameter\_x = Math.round(1.732619659e+31 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(9.900683765e+30 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

fly\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_4();

}

} else {

brain1 = false;

fly\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_4();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

}

function innervate\_1() {

brain\_junction\_1();

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 76640 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 131961 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [(centerpoint\_z + (sizemax \* 0.05 + Math.round((sizemax \* 0.4) / 1.23758547e+30) \* 1.23758547e+30)) + (6.187927353e+26 - 3.093963676e+28), loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.8;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 6.187927353e+29;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.4) - ((Math.floor((sizemin \* 0.09) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - sizemid \* 0.8) - 1.23758547e+30;

motor\_nerve\_x();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.8) - Math.floor((sizemin \* 0.085) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.4) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 76074 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 129031 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [(centerpoint\_z + (sizemax \* 0.05 + Math.round((sizemax \* 0.4) / 1.23758547e+30) \* 1.23758547e+30)) + (6.187927353e+26 - 3.093963676e+28), loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.8;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 6.187927353e+29;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.4) - ((Math.floor((sizemin \* 0.09) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.05) - 1.23758547e+30;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.05) - Math.floor((sizemin \* 0.085) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.4) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 81533 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 137594 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.41, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemid \* 0.78;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_back\_up();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.1 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 1.856378206e+29) - 3.093963676e+28);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - sizemid \* 0.8) - 1.23758547e+30;

motor\_nerve\_x();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.8) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 81429 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 133365 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.42, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemid \* 0.79;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_back\_up();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.29 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 1.856378206e+29) - 3.093963676e+28);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - sizemid \* 0.8) - 1.23758547e+30;

motor\_nerve\_x();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.8) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 78630 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 137112 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.43, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemid \* 0.79;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.49 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 0) - 3.093963676e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - sizemid \* 0.8) - 1.23758547e+30;

motor\_nerve\_x();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.8) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 79352 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 132227 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.44, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemid \* 0.78;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.69 + Math.round((sizemax \* 0.05) / 1.23758547e+30))) + ((6.187927353e+26 + 0) - 3.093963676e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - sizemid \* 0.8) - 1.23758547e+30;

motor\_nerve\_x();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.8) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 85115 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 124115 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.41, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemid \* 0.22;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_back\_up();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.1 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 1.856378206e+29) - 3.093963676e+28);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.1) - 1.23758547e+30;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.1) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 82272 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 129335 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.42, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemid \* 0.21;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_back\_up();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.29 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 1.856378206e+29) - 3.093963676e+28);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.1) - 1.23758547e+30;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.1) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 80308 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 123072 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.43, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemid \* 0.21;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = (centerpoint\_z + (sizemax \* 0.49 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 0) - 3.093963676e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.1) - 1.23758547e+30;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.1) - Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz + 7.425512824e+30, (brainy - (100000 \* 1.23758547e+26 - 79247 \* 1.23758547e+26)) + 3.093963676e+28, (brainx - (60000 \* 1.23758547e+26 + 129109 \* 1.23758547e+26)) + 3.093963676e+28];

loc6 = [centerpoint\_z + sizemax \* 0.44, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemid \* 0.22;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = (centerpoint\_y - sizemin \* 0.2) - ((Math.floor((sizemin \* 0.1) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = (centerpoint\_z + (sizemax \* 0.69 + Math.round((sizemax \* 0.05) / 1.23758547e+30) \* 1.23758547e+30)) + ((6.187927353e+26 + 0) - 3.093963676e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.1) - 1.23758547e+30;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (spawnerx > (centerpoint\_x - sizemid \* 0.1) - Math.floor((sizemid \* 0.1) / 2.475170941e+30)) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

while (loc5[1] < centerpoint\_y - sizemin \* 0.2) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

}

function human\_muscle() {

human\_muscles();

}

function vertebrate\_muscle() {

vertebrate\_muscles();

}

function motor\_unit\_1() {

nerve2 = true;

nerve3 = true;

spawnerz = loc5[0];

spawnery = loc5[1];

spawnerx = loc6[2];

while (nerve2 == true) {

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - cedex];

nerve\_trijunction\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 1.082268494e+30;

motor\_nerve\_y();

motor\_unit\_2();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] + 1.546981839e+26;

cursor2x = loc5[2] - 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

spawnerx = spawnerx - cedex;

loc5 = [spawnerz, spawnery, spawnerx];

loc6 = [spawnerz + cedex, spawnery - cedex, spawnerx - 2.28953312e+30];

motor\_nerve\_x();

loc1 = [loc5[0] + 3.032084403e+28, loc6[1] + 1.23758547e+30, loc6[2] + 1.23758547e+30];

loc2 = [loc5[0] + 1.23758547e+30, loc5[1] - 1.23758547e+30, loc5[2] - 1.23758547e+30];

make = false;

if (bvar == 1) {

vertebrate\_muscle();

} else {

human\_muscle();

}

if (make == false) {

nerve2 = false;

}

}

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc6[2] - 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

}

function motor\_unit\_2() {

nerve3 = true;

while (nerve3 == true) {

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_trijunction\_right();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + 2.28953312e+30;

motor\_nerve\_y();

loc1 = [loc5[0] + 3.032084403e+28, loc6[1] + 1.553169765e+29, loc6[2] + 1.23758547e+30];

loc2 = [loc5[0] + 1.23758547e+30, loc5[1] - 1.23758547e+30, loc5[2] - 1.23758547e+30];

make = false;

if (bvar == 1) {

vertebrate\_muscle();

} else {

human\_muscle();

}

if (make == false) {

nerve3 = false;

}

}

}

function innervate\_2() {

brain\_junction\_2();

sense\_organs\_1();

sensory\_1();

motor\_1();

sensory\_epithelia\_1();

}

function innervate\_3() {

brain\_junction\_3();

sense\_organs\_2();

sensory\_2();

motor\_2();

sensory\_epithelia\_2();

}

function brain\_junction\_2() {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 5.9358 \* 3.093963676e+31, brainx - 6.0272 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 5.9358 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.0272 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.1686 \* 3.093963676e+31, brainx - 5.5574 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.1686 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.5574 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 5.9393 \* 3.093963676e+31, brainx - 6.1896 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 5.9393 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.1896 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.943 \* 3.093963676e+31, brainx - 6.2039 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.943 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.2039 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.1 \* 3.093963676e+31, brainx - 4.55 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.1 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.55 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.785 \* 3.093963676e+31, brainx - 4.575 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.785 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.575 \* 3.093963676e+31) - 1.23758547e+29]);

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.217 \* 3.093963676e+31, brainx - 5.867 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.217 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.867 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.105 \* 3.093963676e+31, brainx - 4.942 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.105 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.942 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.22 \* 3.093963676e+31, brainx - 5.765 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.22 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.765 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.1 \* 3.093963676e+31, brainx - 5.185 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.1 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.185 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.573 \* 3.093963676e+31, brainx - 4.6635 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.573 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.6635 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.124 \* 3.093963676e+31, brainx - 5.14 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.124 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.14 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.4023 \* 3.093963676e+31, brainx - 6.6249 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.4023 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.6249 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.5606 \* 3.093963676e+31, brainx - 4.7942 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.5606 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.7942 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3316 \* 3.093963676e+31, brainx - 5.9411 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3316 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.9411 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.001 \* 3.093963676e+31, brainx - 6.78 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.001 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.78 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.315 \* 3.093963676e+31, brainx - 6.21 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.315 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.21 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.9426 \* 3.093963676e+31, brainx - 6.4006 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.9426 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.4006 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.095 \* 3.093963676e+31, brainx - 6.084 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.095 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.084 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.6439 \* 3.093963676e+31, brainx - 6.8057 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.6439 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.8057 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3226 \* 3.093963676e+31, brainx - 6.301 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3226 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.301 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.8564 \* 3.093963676e+31, brainx - 6.2589 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.8564 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.2589 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 5.9189 \* 3.093963676e+31, brainx - 5.9501 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 5.9189 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.9501 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.9807 \* 3.093963676e+31, brainx - 5.3532 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.9807 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.3532 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.55 \* 3.093963676e+31, brainx - 4.95 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.55 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.95 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.5892 \* 3.093963676e+31, brainx - 5.4981 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.5892 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.4981 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.0702 \* 3.093963676e+31, brainx - 4.957 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.0702 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.957 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.2511 \* 3.093963676e+31, brainx - 4.8975 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.2511 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.8975 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.065 \* 3.093963676e+31, brainx - 4.8 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.065 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.8 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.06 \* 3.093963676e+31, brainx - 4.54 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.06 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.54 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.2895 \* 3.093963676e+31, brainx - 6.4919 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.2895 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.4919 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.86 \* 3.093963676e+31, brainx - 6.28 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.86 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.28 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.315 \* 3.093963676e+31, brainx - 6.265 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.315 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.265 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.948 \* 3.093963676e+31, brainx - 6.31 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.948 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.31 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.13 \* 3.093963676e+31, brainx - 6.125 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.13 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.125 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3 \* 3.093963676e+31, brainx - 6.22 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.22 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.285 \* 3.093963676e+31, brainx - 5.45 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.285 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.45 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.077 \* 3.093963676e+31, brainx - 5.096 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.077 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.096 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.265 \* 3.093963676e+31, brainx - 5.345 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.265 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.345 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.879 \* 3.093963676e+31, brainx - 5.0579 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.879 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.0579 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.043 \* 3.093963676e+31, brainx - 4.9286 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.043 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.9286 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.6639 \* 3.093963676e+31, brainx - 5.3998 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.6639 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.3998 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.856378206e+29 \* 1;

density\_parameter\_y = 1.856378206e+29 \* 1;

cursor1x = brainx - (3.712756412e+32 / 2) \* 1;

cursor1y = brainy - ((3.384796262e+32 / 2) \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 0);

generation\_parameter\_x = Math.ceil((2.165774573e+31 / density\_parameter\_x) / 2) \* 2;

generation\_parameter\_3 = Math.ceil((2.165774573e+31 / density\_parameter\_y) / 2) \* 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

list\_tissues.push(['nervous', brainz + (3.520930664e+32 - 6.187927353e+26), cursor1y, cursor1x, brainz + (3.520930664e+32 + 6.187927353e+26), cursor1y - 1.23758547e+29, cursor1x - 1.23758547e+29]);

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.856378206e+29 \* 1;

density\_parameter\_y = 1.856378206e+29 \* 1;

cursor1x = brainx - (2.908325856e+32 / 2) \* 1;

cursor1y = brainy - ((3.384796262e+32 / 2) \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 0);

generation\_parameter\_x = Math.ceil((2.165774573e+31 / density\_parameter\_x) / 2) \* 2;

generation\_parameter\_3 = Math.ceil((2.165774573e+31 / density\_parameter\_y) / 2) \* 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

list\_tissues.push(['nervous', brainz + (3.520930664e+32 - 6.187927353e+26), cursor1y, cursor1x, brainz + (3.520930664e+32 + 6.187927353e+26), cursor1y - 1.23758547e+29, cursor1x - 1.23758547e+29]);

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - (0 \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 1.546981839e+26);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_3();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_3();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - 0 \* 1;

cursor1z = brainz + (3.520930664e+32 \* 1 - 0);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_4();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_4();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - (0 \* 1 - 0);

cursor1z = brainz + (0 - 1.546981839e+26);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_3();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_3();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - 0 \* 1;

cursor1z = brainz + (0 \* 1 - 0);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_4();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_4();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

}

function brain\_junction\_3() {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.185 \* 3.093963676e+31, brainx - 5.76 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.185 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.76 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.5093 \* 3.093963676e+31, brainx - 5.86 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.5063 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.86 \* 3.093963676e+31) - 1.23758547e+29]);

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.217 \* 3.093963676e+31, brainx - 5.867 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.217 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.867 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.105 \* 3.093963676e+31, brainx - 4.942 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.105 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.942 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.22 \* 3.093963676e+31, brainx - 5.765 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.22 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.765 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.1 \* 3.093963676e+31, brainx - 5.185 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.1 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.185 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.573 \* 3.093963676e+31, brainx - 4.6635 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.573 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.6635 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.124 \* 3.093963676e+31, brainx - 5.14 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.124 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.14 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.4023 \* 3.093963676e+31, brainx - 6.6249 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.4023 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.6249 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.5606 \* 3.093963676e+31, brainx - 4.7942 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.5606 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.7942 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3316 \* 3.093963676e+31, brainx - 5.9411 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3316 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.9411 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.001 \* 3.093963676e+31, brainx - 6.78 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.001 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.78 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.315 \* 3.093963676e+31, brainx - 6.21 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.315 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.21 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.9426 \* 3.093963676e+31, brainx - 6.4006 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.9426 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.4006 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.095 \* 3.093963676e+31, brainx - 6.084 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.095 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.084 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.6439 \* 3.093963676e+31, brainx - 6.8057 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.6439 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.8057 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3226 \* 3.093963676e+31, brainx - 6.301 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3226 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.301 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.8564 \* 3.093963676e+31, brainx - 6.2589 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.8564 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.2589 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 5.9189 \* 3.093963676e+31, brainx - 5.9501 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 5.9189 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.9501 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.9807 \* 3.093963676e+31, brainx - 5.3532 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.9807 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.3532 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.55 \* 3.093963676e+31, brainx - 4.95 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.55 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.95 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.5892 \* 3.093963676e+31, brainx - 5.4981 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.5892 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.4981 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.0702 \* 3.093963676e+31, brainx - 4.957 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.0702 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.957 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.2511 \* 3.093963676e+31, brainx - 4.8975 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.2511 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.8975 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.065 \* 3.093963676e+31, brainx - 4.8 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.065 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.8 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.06 \* 3.093963676e+31, brainx - 4.54 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.06 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.54 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.2895 \* 3.093963676e+31, brainx - 6.4919 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.2895 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.4919 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.86 \* 3.093963676e+31, brainx - 6.28 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.86 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.28 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.315 \* 3.093963676e+31, brainx - 6.265 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.315 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.265 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.948 \* 3.093963676e+31, brainx - 6.31 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.948 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.31 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.13 \* 3.093963676e+31, brainx - 6.125 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.13 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.125 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.3 \* 3.093963676e+31, brainx - 6.22 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.3 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 6.22 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.285 \* 3.093963676e+31, brainx - 5.45 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.285 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.45 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 7.077 \* 3.093963676e+31, brainx - 5.096 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 7.077 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.096 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.265 \* 3.093963676e+31, brainx - 5.345 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.265 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.345 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.879 \* 3.093963676e+31, brainx - 5.0579 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.879 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.0579 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.043 \* 3.093963676e+31, brainx - 4.9286 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.043 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 4.9286 \* 3.093963676e+31) - 1.23758547e+29]);

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), brainy - 6.6639 \* 3.093963676e+31, brainx - 5.3998 \* 3.093963676e+31, brainz + (0 + 6.187927353e+26), (brainy - 6.6639 \* 3.093963676e+31) - 1.23758547e+29, (brainx - 5.3998 \* 3.093963676e+31) - 1.23758547e+29]);

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.856378206e+29 \* 1;

density\_parameter\_y = 1.856378206e+29 \* 1;

cursor1x = brainx - (3.712756412e+32 / 2) \* 1;

cursor1y = brainy - ((3.384796262e+32 / 2) \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 0);

generation\_parameter\_x = Math.ceil((2.165774573e+31 / density\_parameter\_x) / 2) \* 2;

generation\_parameter\_3 = Math.ceil((2.165774573e+31 / density\_parameter\_y) / 2) \* 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

list\_tissues.push(['nervous', brainz + (3.520930664e+32 - 6.187927353e+26), cursor1y, cursor1x, brainz + (3.520930664e+32 + 6.187927353e+26), cursor1y - 1.23758547e+29, cursor1x - 1.23758547e+29]);

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.856378206e+29 \* 1;

density\_parameter\_y = 1.856378206e+29 \* 1;

cursor1x = brainx - (2.908325856e+32 / 2) \* 1;

cursor1y = brainy - ((3.384796262e+32 / 2) \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 0);

generation\_parameter\_x = Math.ceil((2.165774573e+31 / density\_parameter\_x) / 2) \* 2;

generation\_parameter\_3 = Math.ceil((2.165774573e+31 / density\_parameter\_y) / 2) \* 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

list\_tissues.push(['nervous', brainz + (3.520930664e+32 - 6.187927353e+26), cursor1y, cursor1x, brainz + (3.520930664e+32 + 6.187927353e+26), cursor1y - 1.23758547e+29, cursor1x - 1.23758547e+29]);

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.856378206e+29 \* 1;

density\_parameter\_y = 1.856378206e+29 \* 1;

cursor1x = brainx - (2.405061724e+32 / 2) \* 1;

cursor1y = brainy - ((3.476810742e+32 / 2) \* 1 - 0);

cursor1z = brainz;

generation\_parameter\_x = Math.ceil((2.165774573e+31 / density\_parameter\_x) / 2) \* 2;

generation\_parameter\_3 = Math.ceil((2.165774573e+31 / density\_parameter\_y) / 2) \* 2;

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

list\_tissues.push(['nervous', brainz + (0 - 6.187927353e+26), cursor1y, cursor1x, brainz + (0 + 6.187927353e+26), cursor1y - 1.23758547e+29, cursor1x - 1.23758547e+29]);

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - (0 \* 1 - 0);

cursor1z = brainz + (3.520930664e+32 - 1.546981839e+26);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_3();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_3();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - 0 \* 1;

cursor1z = brainz + (3.520930664e+32 \* 1 - 0);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_4();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_4();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - (0 \* 1 - 0);

cursor1z = brainz + (0 - 1.546981839e+26);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_3();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_3();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = brainx - 0 \* 1;

cursor1y = brainy - 0 \* 1;

cursor1z = brainz + (0 \* 1 - 0);

generation\_parameter\_x = Math.round(3.527118591e+32 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.475170941e+32 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.520930664e+32 / density\_parameter\_z - 0);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc1 = [cursor1z + 1.546981837e+26, cursor1y - 4.085906751e+25, cursor1x - 4.085906751e+25];

make = false;

target1 = ['nervous'];

find2();

if (make == true) {

brain2 = false;

mouse\_brain\_in();

if (brain2 == true) {

randomize\_lipid\_4();

}

} else {

brain1 = false;

mouse\_brain\_in();

if (brain1 == true) {

randomize\_lipid\_4();

}

}

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

}

function sense\_organs\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+28;

cedey = 6.187927353e+28;

cedez = 6.187927353e+28;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.213) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 0.9) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.213) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + ((centerpoint\_z + sizemax \* 0.9) - spawnerz)) < (sizemin \* 0.0148125) \* 0.1) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.213) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1.2) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.213) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

if (spawnery < centerpoint\_y - sizemin \* 0.5025) {

if (spawnery > centerpoint\_y - sizemin \* 0.4975) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.787) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 0.9) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.787) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + ((centerpoint\_z + sizemax \* 0.9) - spawnerz)) < (sizemin \* 0.0148125) \* 0.1) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.787) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1.2) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.787) - spawnerx, 2))) < (sizemin \* 0.0148125) \* 1.4) {

if (Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - spawnerx, 2)) > (sizemin \* 0.3) \* 1) {

if (spawnery < centerpoint\_y - sizemin \* 0.5025) {

if (spawnery > centerpoint\_y - sizemin \* 0.4975) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

eye\_lenses\_1();

}

function sense\_organs\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+28;

cedey = 6.187927353e+28;

cedez = 6.187927353e+28;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 0.9) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_x - sizemid \* 0.45) - spawnerx, 2) + ((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz)) < (sizemax \* 0.0148125) \* 0.1) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1.3) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1.5) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

if (spawnery < centerpoint\_y - sizemin \* 0.9308) {

if (spawnery > centerpoint\_y - sizemin \* 0.931) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 0.9) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1) {

if (Math.sqrt(Math.pow((centerpoint\_x - sizemid \* 0.55) - spawnerx, 2) + ((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz)) < (sizemax \* 0.0148125) \* 0.1) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

list\_tissues.push(['eye1', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1.3) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - spawnerz, 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - spawnery, 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - spawnerx, 2))) < ((sizemax \* 0.0148125) / 2) \* 1.5) {

if (spawnery < centerpoint\_y - sizemin \* 0.38) {

if (spawnery < centerpoint\_y - sizemin \* 0.9308) {

if (spawnery > centerpoint\_y - sizemin \* 0.931) {

list\_tissues.push(['eye2', spawnerz, spawnery, spawnerx, spawnerz + cedez, spawnery - cedey, spawnerx - cedex]);

}

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

eye\_lenses\_2();

list\_tissues.push(['ear', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0, centerpoint\_z + sizemax \* 0.95666, centerpoint\_y - sizemin \* 0.21, centerpoint\_x - sizemid \* 0.252]);

list\_tissues.push(['ear', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.2005, centerpoint\_x - sizemid \* 0.2523, centerpoint\_z + sizemax \* 0.95666, centerpoint\_y - sizemin \* 0.21, centerpoint\_x - sizemid \* 0.258]);

list\_tissues.push(['ear', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.2523, centerpoint\_z + sizemax \* 0.95666, centerpoint\_y - sizemin \* 0.1995, centerpoint\_x - sizemid \* 0.258]);

list\_tissues.push(['ear', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.1995, centerpoint\_x - sizemid \* 0.2523, centerpoint\_z + sizemax \* 0.9532, centerpoint\_y - sizemin \* 0.2005, centerpoint\_x - sizemid \* 0.258]);

list\_tissues.push(['ear', centerpoint\_z + sizemax \* 0.9534, centerpoint\_y - sizemin \* 0.1995, centerpoint\_x - sizemid \* 0.2523, centerpoint\_z + sizemax \* 0.95666, centerpoint\_y - sizemin \* 0.2005, centerpoint\_x - sizemid \* 0.258]);

list\_tissues.push(['cochlea', centerpoint\_z + sizemax \* 0.95, centerpoint\_y - sizemin \* 0.19, centerpoint\_x - sizemid \* 0.2583, centerpoint\_z + sizemax \* 0.95666, centerpoint\_y - sizemin \* 0.21, centerpoint\_x - sizemid \* 0.267]);

}

function eye\_lenses\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemid / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.203) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x > centerpoint\_x - sizemid \* 0.2005) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) < sizemin \* 0.007) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.797) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.5) - cursor1y, 2))) > sizemin \* 0.0065) {

if (cursor1x < centerpoint\_x - sizemid \* 0.7995) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function eye\_lenses\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1x = cursor1x + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 2;

cursor1y = cursor1y + density\_parameter\_x / 1;

cursor1x = cursor1x + density\_parameter\_x / 2;

}

sorter = 14;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x / 4;

cursor1x = cursor1x + (density\_parameter\_x - 0.75);

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x / 4;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x \* 0.75;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemid / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

if (unused\_variable == 5) {

cursor1z = cursor1z + density\_parameter\_x \* 0.75;

cursor1y = cursor1y + density\_parameter\_x \* 0.75;

cursor1x = cursor1x + density\_parameter\_x / 4;

}

sorter = 15;

density\_parameter\_x = 2.207233687e+25;

density\_parameter\_y = 2.207233687e+25;

density\_parameter\_z = 2.207233687e+25;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x);

generation\_parameter\_y = Math.round(sizemin / density\_parameter\_y);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

cursor1x = cursor1x - density\_parameter\_x;

for (countcg2 = 0; countcg2 < generation\_parameter\_y; countcg2++) {

cursor1y = cursor1y - density\_parameter\_y;

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

cursor1z = cursor1z + density\_parameter\_z;

set\_earth\_rotation();

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.45) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

} else if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) < sizemin \* 0.012) {

if (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - cursor1z, 2) + (Math.pow((centerpoint\_x - sizemid \* 0.55) - cursor1x, 2) + Math.pow((centerpoint\_y - sizemin \* 0.39) - cursor1y, 2))) > sizemin \* 0.011) {

if (cursor1y < centerpoint\_y - sizemid \* 0.393) {

sort\_allotrope\_bar();

}

}

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1y = cursor1y + generation\_parameter\_y \* density\_parameter\_y;

}

}

}

function sensory\_1() {

if (unused\_variable == 5) {

spawnerx2 = brainx - (3.712756412e+32 / 2 - 3.093963676e+28);

spawnery2 = brainy - (3.384796262e+32 / 2 - 3.093963676e+28);

spawnerz2 = brainz + 3.520930664e+32;

cedex = 1.856378206e+29;

cedey = 1.856378206e+29;

cedez = 1.856378206e+29;

repeat\_x = Math.ceil(((4.331549147e+31 / 2) / cedex) / 2) \* 2;

repeat\_y = repeat\_x;

nerve = [repeat\_x, 0];

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

if (unused\_variable == 5) {

loc5 = [spawnerz2, spawnery2, spawnerx2];

loc6 = [(centerpoint\_z + sizemax \* 0.9) + (nerve[1] \* cedex - (cedex \* repeat\_x) / 2), spawnery2 - cedey, spawnerx2 - cedex];

nerve\_z();

loc5 = [loc6[0], spawnery2, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2 - cedey, spawnerx2 - cedex];

nerve\_junction\_back\_down();

loc5 = [loc6[0] - cedex, (centerpoint\_y - (sizemin \* 0.5 + (cedex \* repeat\_x) / 2)) + nerve[0] \* cedex, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2, spawnerx2 - cedex];

nerve\_y();

loc6 = [loc5[0] + cedex, loc5[1], spawnerx2 - cedex];

loc5 = [loc6[0] - cedex, loc6[1] + cedex, spawnerx2];

nerve\_junction\_up\_left();

loc5 = [loc6[0] - cedex, loc6[1] + cedex, loc6[2]];

loc6 = [loc5[0] + cedex, loc5[1] - cedex, loc5[2] - (cedex \* repeat\_x + (nerve[1] \* cedex - cedex \* Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0]))))];

nerve\_x();

loc5[loc5.length - 1] = loc6[2];

loc6[loc6.length - 1] = loc5[2] - cedex;

if (nerve[0] > repeat\_x / 2) {

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex) + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)));

nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

} else {

nerve\_junction\_up\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

}

} else {

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex) + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)));

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

} else {

nerve\_junction\_down\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

}

}

nerve2 = true;

while (nerve2 == true && (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - loc5[0], 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.787) - loc5[2], 2))) > (sizemin \* 0.0148125) \* 0.9 + 2.475170941e+30 && Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc5[2], 2)) < sizemin \* 0.3)) {

loc5[loc5.length - 1] = loc6[2] - 1;

loc6[loc6.length - 1] = loc5[2] - 6.187927353e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_cells();

screen\_molecules();

if (make != true) {

nerve2 = false;

cursor2x = loc6[2] - 1.546981839e+26;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

} else {

nerve\_x();

}

}

if (nerve2 == true) {

loc5[loc5.length - 1] = loc6[2] - 1;

loc6[loc6.length - 1] = loc5[2] - 1.856378206e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_molecules();

screen\_cells();

if (make != true) {

nerve2 = false;

cursor2x = loc6[2] - 1.546981839e+26;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

} else {

nerve\_junction\_right\_forth();

}

}

if (nerve2 == true) {

make = true;

loc1 = [loc6[0] + 1, loc5[1] + 2.165774573e+29, loc5[2] + 2.165774573e+29];

loc2 = [loc2[0] + 6.187927352e+30, loc2[1] - 6.187927352e+29, loc2[2] - 6.187927352e+29];

screen\_molecules();

screen\_cells();

if (make == true) {

spawnerz = loc1[0];

spawnery = loc1[1];

spawnerx = loc1[2];

rod\_cell();

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

}

}

spawnery2 = spawnery2 - cedey;

if (unused\_variable == 5) {

nerve3 = nerve[1];

nerve[1] = nerve3 + 1;

}

}

spawnerx2 = spawnerx2 - cedex;

spawnery2 = spawnery2 + repeat\_y \* cedex;

if (unused\_variable == 5) {

nerve3 = nerve[0];

nerve[0] = nerve3 - 1;

}

}

}

if (unused\_variable == 5) {

spawnerx2 = brainx - (2.908325856e+32 / 2 - 3.093963676e+28);

spawnery2 = brainy - (3.384796262e+32 / 2 - 3.093963676e+28);

spawnerz2 = brainz + 3.520930664e+32;

cedex = 1.856378206e+29;

cedey = 1.856378206e+29;

cedez = 1.856378206e+29;

repeat\_x = Math.ceil(((4.331549147e+31 / 2) / cedex) / 2) \* 2;

repeat\_y = repeat\_x;

nerve = [0, 0];

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

if (unused\_variable == 5) {

loc5 = [spawnerz2, spawnery2, spawnerx2];

loc6 = [(centerpoint\_z + sizemax \* 0.9) + (nerve[1] \* cedex - (cedex \* repeat\_x) / 2), spawnery2 - cedey, spawnerx2 - cedex];

nerve\_z();

loc5 = [loc6[0], spawnery2, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2 - cedey, spawnerx2 - cedex];

nerve\_junction\_back\_down();

loc5 = [loc6[0] - cedex, (centerpoint\_y - (sizemin \* 0.5 - (cedex \* repeat\_x) / 2)) + nerve[0] \* cedex, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2, spawnerx2 - cedex];

nerve\_y();

loc6 = [loc5[0] + cedex, loc5[1], spawnerx2 - cedex];

loc5 = [loc6[0] - cedex, loc6[1] + cedex, spawnerx2];

nerve\_junction\_right\_up();

loc6 = [loc5[0] + cedex, loc5[1] - cedex, loc5[2]];

loc5 = [loc6[0] - cedex, loc6[1] + cedex, loc6[2] - (cedex \* repeat\_x + (nerve[1] \* cedex - cedex \* Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0]))))];

nerve\_x();

loc6[loc6.length - 1] = loc5[2];

loc5[loc5.length - 1] = loc6[2] + cedex;

if (nerve[0] > repeat\_x / 2) {

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex) + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)));

nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

} else {

nerve\_junction\_up\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

}

} else {

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex) + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)));

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

} else {

nerve\_junction\_down\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((sizemin \* 0.0148125) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

}

}

nerve2 = true;

while (nerve2 == true && (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.9) - loc5[0], 2) + (Math.pow((centerpoint\_y - sizemin \* 0.5) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.213) - loc5[2], 2))) > (sizemin \* 0.0148125) \* 0.9 + 2.475170941e+30 && Math.sqrt(Math.pow((centerpoint\_y - sizemin \* 0.5) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.5) - loc5[2], 2)) < sizemin \* 0.3)) {

loc6[loc6.length - 1] = loc5[2] + 1;

loc5[loc5.length - 1] = loc6[2] + 6.187927353e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_cells();

screen\_molecules();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] + 1.546981839e+26;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

} else {

nerve\_x();

}

}

if (nerve2 == true) {

loc5[loc5.length - 1] = loc6[2] - 1;

loc6[loc6.length - 1] = loc5[2] - 1.856378206e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_molecules();

screen\_cells();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] + 1.546981839e+26;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

} else {

nerve\_junction\_left\_forth();

}

}

if (nerve2 == true) {

make = true;

loc1 = [loc6[0] + 1, loc5[1] + 2.165774573e+29, loc5[2] + 2.165774573e+29];

loc2 = [loc2[0] + 6.187927352e+30, loc2[1] - 6.187927352e+29, loc2[2] - 6.187927352e+29];

screen\_molecules();

screen\_cells();

if (make == true) {

spawnerz = loc1[0];

spawnery = loc1[1];

spawnerx = loc1[2];

rod\_cell();

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

}

}

spawnery2 = spawnery2 - cedey;

if (unused\_variable == 5) {

nerve3 = nerve[1];

nerve[1] = nerve3 + 1;

}

}

spawnerx2 = spawnerx2 - cedex;

spawnery2 = spawnery2 + repeat\_y \* cedex;

if (unused\_variable == 5) {

nerve3 = nerve[0];

nerve[0] = nerve3 + 1;

}

}

}

}

function sensory\_2() {

if (unused\_variable == 5) {

spawnerx2 = brainx - (3.712756412e+32 / 2 - 3.093963676e+28);

spawnery2 = brainy - (3.384796262e+32 / 2 - 3.093963676e+28);

spawnerz2 = brainz + 3.520930664e+32;

cedex = 1.856378206e+29;

cedey = 1.856378206e+29;

cedez = 1.856378206e+29;

repeat\_x = Math.ceil(((4.331549147e+31 / 2) / cedex) / 2) \* 2;

repeat\_y = repeat\_x;

nerve = [repeat\_x, repeat\_x];

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

if (unused\_variable == 5) {

loc5 = [spawnerz2, spawnery2, spawnerx2];

loc6 = [(spawnerz2 + sizemax \* 0.01) + (nerve[1] \* cedex - (cedex \* repeat\_x) / 2), spawnery2 - cedey, spawnerx2 - cedex];

nerve\_z();

loc5 = [loc6[0], spawnery2, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2 - cedey, spawnerx2 - cedex];

nerve\_junction\_back\_up();

loc5 = [loc6[0] - cedex, loc6[0], spawnerx2];

loc6 = [loc5[0] + cedex, (brainy - (sizemin \* 0.01 + (cedex \* repeat\_x) / 2)) - nerve[1] \* cedex, spawnerx2 - cedex];

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.93103448275 + (cedex \* repeat\_x) / 2)) - cedex \* nerve[0];

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = (centerpoint\_x - (sizemid \* 0.55 - (cedex \* repeat\_x) / 2)) - cedex \* (repeat\_x - nerve[1]);

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) \* cedex) + (cedex \* repeat\_x + (cedex \* repeat\_x - nerve[1] \* cedex)));

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

if (nerve[0] > repeat\_x / 2) {

nerve\_junction\_down\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)) + (repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

} else {

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

}

} else {

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)) + (repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

} else {

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

}

}

nerve2 = true;

while (nerve2 == true && (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc5[0], 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.55) - loc5[2], 2))) > 2.475170941e+30 + ((sizemax \* 0.0148125) / 2) \* 0.9 && loc5[1] > centerpoint\_y - sizemin \* 0.38)) {

loc5[1] = loc6[1] - 1;

loc6[1] = loc5[1] - 6.187927353e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_cells();

screen\_molecules();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc6[1] - 1.546981839e+26;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

} else {

nerve\_y();

}

}

if (nerve2 == true) {

loc5[loc5.length - 1] = loc6[1] - 1;

loc6[loc6.length - 1] = loc5[1] - 1.856378206e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_molecules();

screen\_cells();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc6[1] - 1.546981839e+26;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

} else {

nerve\_junction\_down\_forth();

}

}

if (nerve2 == true) {

make = true;

loc1 = [loc6[0] + 1, loc5[1] + 2.165774573e+29, loc5[2] + 2.165774573e+29];

loc2 = [loc2[0] + 6.187927352e+30, loc2[1] - 6.187927352e+29, loc2[2] - 6.187927352e+29];

screen\_molecules();

screen\_cells();

if (make == true) {

spawnerz = loc1[0];

spawnery = loc1[1];

spawnerx = loc1[2];

rod\_cell();

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

}

}

spawnery2 = spawnery2 - cedey;

if (unused\_variable == 5) {

nerve3 = nerve[1];

nerve[1] = nerve3 - 1;

}

}

spawnerx2 = spawnerx2 - cedex;

spawnery2 = spawnery2 + repeat\_y \* cedex;

if (unused\_variable == 5) {

nerve3 = nerve[0];

nerve[0] = nerve3 - 1;

}

}

}

if (unused\_variable == 5) {

spawnerx2 = brainx - (2.908325856e+32 / 2 - 3.093963676e+28);

spawnery2 = brainy - (3.384796262e+32 / 2 - 3.093963676e+28);

spawnerz2 = brainz + 3.520930664e+32;

cedex = 1.856378206e+29;

cedey = 1.856378206e+29;

cedez = 1.856378206e+29;

repeat\_x = Math.ceil(((4.331549147e+31 / 2) / cedex) / 2) \* 2;

repeat\_y = repeat\_x;

nerve = [0, repeat\_x];

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

if (unused\_variable == 5) {

loc5 = [spawnerz2, spawnery2, spawnerx2];

loc6 = [(spawnerz2 + sizemax \* 0.01) + (nerve[1] \* cedex - (cedex \* repeat\_x) / 2), spawnery2 - cedey, spawnerx2 - cedex];

nerve\_z();

loc5 = [loc6[0], spawnery2, spawnerx2];

loc6 = [loc5[0] + cedex, spawnery2 - cedey, spawnerx2 - cedex];

nerve\_junction\_back\_up();

loc5 = [loc6[0] - cedex, loc6[0], spawnerx2];

loc6 = [loc5[0] + cedex, (brainy - (sizemin \* 0.01 + (cedex \* repeat\_x) / 2)) - nerve[1] \* cedex, spawnerx2 - cedex];

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = (centerpoint\_z + (sizemax \* 0.93103448275 + (cedex \* repeat\_x) / 2)) - cedex \* nerve[0];

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - (sizemid \* 0.45 - (cedex \* repeat\_x) / 2)) + cedex \* (repeat\_x - nerve[1]);

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - ((repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) \* cedex) + (cedex \* repeat\_x + (cedex \* repeat\_x - nerve[1] \* cedex)));

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

if (nerve[0] > repeat\_x / 2) {

nerve\_junction\_down\_back();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - ((((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)) + (repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex));

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

} else {

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

}

} else {

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + ((((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[0])) / (repeat\_x / 2)) + (repeat\_y \* cedex - Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) \* cedex));

nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

if (nerve[1] > repeat\_x / 2) {

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

} else {

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - (((sizemax \* 0.0148125) / 2) \* 0.9) \* (Math.abs(repeat\_x / 2 - (repeat\_x - nerve[1])) / (repeat\_x / 2));

nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_up();

}

}

nerve2 = true;

while (nerve2 == true && (Math.sqrt(Math.pow((centerpoint\_z + sizemax \* 0.93103448275) - loc5[0], 2) + (Math.pow((centerpoint\_y - sizemin \* 0.3755) - loc5[1], 2) + Math.pow((centerpoint\_x - sizemid \* 0.45) - loc5[2], 2))) > 2.475170941e+30 + ((sizemax \* 0.0148125) / 2) \* 0.9 && loc5[1] > centerpoint\_y - sizemin \* 0.38)) {

loc5[1] = loc6[1] - 1;

loc6[1] = loc5[1] - 6.187927353e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_cells();

screen\_molecules();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc6[1] - 1.546981839e+26;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

} else {

nerve\_y();

}

}

if (nerve2 == true) {

loc5[loc5.length - 1] = loc6[1] - 1;

loc6[loc6.length - 1] = loc5[1] - 1.856378206e+29;

make = true;

loc1 = loc5;

loc2 = loc6;

screen\_molecules();

screen\_cells();

if (make != true) {

nerve2 = false;

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc6[1] - 1.546981839e+26;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

} else {

nerve\_junction\_down\_forth();

}

}

if (nerve2 == true) {

make = true;

loc1 = [loc6[0] + 1, loc5[1] + 2.165774573e+29, loc5[2] + 2.165774573e+29];

loc2 = [loc2[0] + 6.187927352e+30, loc2[1] - 6.187927352e+29, loc2[2] - 6.187927352e+29];

screen\_molecules();

screen\_cells();

if (make == true) {

spawnerz = loc1[0];

spawnery = loc1[1];

spawnerx = loc1[2];

rod\_cell();

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

}

}

spawnery2 = spawnery2 - cedey;

if (unused\_variable == 5) {

nerve3 = nerve[1];

nerve[1] = nerve3 - 1;

}

}

spawnerx2 = spawnerx2 - cedex;

spawnery2 = spawnery2 + repeat\_y \* cedex;

if (unused\_variable == 5) {

nerve3 = nerve[0];

nerve[0] = nerve3 + 1;

}

}

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

cedey = 1.856378206e+29;

cedez = 1.856378206e+29;

repeat\_x = Math.ceil(((4.331549147e+31 / 2) / cedex) / 2) \* 2;

repeat\_y = repeat\_x;

eara = 0;

nerve = [0, repeat\_x];

spawnerx2 = brainx - (2.405061724e+32 / 2 - (repeat\_x \* cedex + 3.093963676e+28));

spawnery2 = brainy - (3.476810742e+32 / 2 - 3.093963676e+28);

spawnerz2 = brainz + 0;

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

if (unused\_variable == 5) {

loc6 = [spawnerz2, spawnery2 - cedey, spawnerx2 - cedex];

loc5 = [brainz - (cedex \* repeat\_x + cedex \* nerve[0]), spawnery2, spawnerx2];

nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.4) + cedex \* nerve[1];

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = (centerpoint\_y - sizemin \* 0.19) - cedex \* (repeat\_x - nerve[1]);

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = (centerpoint\_x - sizemid \* 0.267) + ((sizemid \* 0.007) \* (nerve[1] / repeat\_x) + cedex \* nerve[0]);

nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[0] = loc5[1] - (cedex + (sizemin \* 0.17) \* (nerve[0] / repeat\_x));

nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = (centerpoint\_z + sizemax \* 0.95) - 1.546981838e+30;

loc1 = loc5;

loc2 = loc6;

make = true;

screen\_cells();

screen\_molecules();

if (make == true) {

nerve\_z();

make = true;

loc1 = [loc6[0] + 1, loc5[1] + 2.165774573e+29, loc5[2] + 2.165774573e+29];

loc2 = [loc2[0] + (2.784567309e+30 + eara), loc2[1] - 6.187927352e+29, loc2[2] - 6.187927352e+29];

screen\_molecules();

screen\_cells();

if (make == true) {

spawnerz = loc1[0];

spawnery = loc1[1];

spawnerx = loc1[2];

audiocyte();

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

} else {

cursor2x = loc5[2] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc6[0] + 1.546981839e+26;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_z();

}

}

spawnery2 = spawnery2 - cedey;

if (unused\_variable == 5) {

eara = 2.475170941e+30 \* (nerve[1] / repeat\_x);

nerve3 = nerve[1];

nerve[1] = nerve3 - 1;

}

}

spawnerx2 = spawnerx2 - cedex;

spawnery2 = spawnery2 + repeat\_y \* cedex;

if (unused\_variable == 5) {

nerve3 = nerve[0];

nerve[0] = nerve3 + 1;

}

}

}

}

function sensory\_epithelia\_1() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

melanocyte();

} else if (randomizer <= 30) {

melanoblast();

} else if (randomizer <= 40) {

ophalmoblast();

} else if (randomizer <= 50) {

neuroblast();

} else {

ophalmocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye2'];

layer();

if (make == true) {

epiophalmocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.5 + 1.243773398e+30 \* 2);

spawnerz = centerpoint\_z + sizemax \* 0.95;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

while (spawnerz < centerpoint\_z + sizemax \* 0.97) {

spawnerz = spawnerz + cedez;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

olfactocyte();

} else {

neuroblast();

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.38 - 1.243773398e+30 \* 2);

spawnerz = centerpoint\_z + sizemax \* 0.8;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

while (spawnerz < centerpoint\_z + sizemax \* 0.85) {

spawnerz = spawnerz + cedez;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

gustocyte();

} else {

neuroblast();

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1', 'eye2', 'cochlea'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

vertebrate\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1', 'eye2', 'cochlea'];

layer();

if (make == true) {

make = false;

vertebrate\_body();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function sensory\_epithelia\_2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

melanocyte();

} else if (randomizer <= 30) {

melanoblast();

} else if (randomizer <= 40) {

ophalmoblast();

} else if (randomizer <= 50) {

neuroblast();

} else {

ophalmocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye2'];

layer();

if (make == true) {

epiophalmocyte();

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x;

spawnery = centerpoint\_y;

spawnerz = centerpoint\_z;

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

for (countcg1 = 0; countcg1 < repeat\_x; countcg1++) {

for (countcg2 = 0; countcg2 < repeat\_y; countcg2++) {

for (countcg3 = 0; countcg3 < repeat\_z; countcg3++) {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187926734e+29, spawnery - 6.187926734e+29, spawnerx - 6.187926734e+29];

make = true;

screen\_tissues();

screen\_cells();

screen\_molecules();

if (make == true) {

make = false;

thickabs = 1.243773398e+30;

thickabs2 = 0;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['cochlea'];

layer();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

audioblast();

} else if (randomizer <= 40) {

neuroblast();

} else {

otocyte();

}

}

}

spawnerz = spawnerz + cedez;

}

spawnery = spawnery - cedey;

spawnerz = spawnerz - repeat\_z \* cedez;

}

spawnerx = spawnerx - cedex;

spawnery = spawnery + repeat\_y \* cedey;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - sizemin \* 0.24;

spawnerz = centerpoint\_z + (sizemax \* 0.93 + 1.243773398e+30 \* 2);

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

while (spawnery > centerpoint\_y - sizemin \* 0.3) {

spawnery = spawnery - cedey;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

olfactocyte();

} else {

neuroblast();

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - sizemin \* 0.3;

spawnerz = centerpoint\_z + (sizemax \* 0.915 - 1.243773398e+30 \* 2);

}

cedex = 6.187927353e+29;

cedey = 6.187927353e+29;

cedez = 6.187927353e+29;

repeat\_x = Math.round(sizemid / cedex - 0);

repeat\_y = Math.round(sizemin / cedey - 0);

repeat\_z = Math.round(sizemax / cedez - 0);

while (spawnery > centerpoint\_y - sizemin \* 0.35) {

spawnery = spawnery - cedey;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

gustocyte();

} else {

neuroblast();

}

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1', 'eye2', 'cochlea'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = centerpoint\_x;

cursor1y = centerpoint\_y;

cursor1z = centerpoint\_z;

}

density\_parameter\_x = 1.033383868e+26 \* 2;

density\_parameter\_y = 1.033383868e+26 \* 2;

density\_parameter\_z = 3.279601497e+28;

cursor1x = centerpoint\_x - 1.033383868e+26 \* 1;

cursor1y = centerpoint\_y - 1.033383868e+26 \* 1;

cursor1z = centerpoint\_z + 3.279601497e+28 / 2;

generation\_parameter\_x = Math.round(sizemid / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(sizemin / density\_parameter\_y - 2);

generation\_parameter\_z = Math.round(sizemax / density\_parameter\_z - 2);

for (countcg1 = 0; countcg1 < generation\_parameter\_x; countcg1++) {

for (countcg2 = 0; countcg2 < generation\_parameter\_3; countcg2++) {

for (countcg3 = 0; countcg3 < generation\_parameter\_z; countcg3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

make = false;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

human\_body();

if (make == true) {

exempt = [null];

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

set\_earth\_rotation();

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + 3.279601495e+28, cursor1y - 1.033383864e+26, cursor1x - 1.033383864e+26];

make = false;

thickabs = 1.243773398e+30 + 3.279601497e+28 \* 2.1;

thickabs2 = 1.243773398e+30;

thickbod = null;

thickbod2 = null;

thickrel = null;

thickrel2 = null;

target2 = null;

target1 = ['eye1', 'eye2', 'cochlea'];

layer();

if (make == true) {

make = false;

human\_body();

if (make == true) {

collagen();

}

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1y = cursor1y - density\_parameter\_y;

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

}

cursor1x = cursor1x - density\_parameter\_x;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

}

function motor\_2() {

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = centerpoint\_y - (sizemin \* 0.02 + (2.475170941e+32 + 6.187927353e+29));

spawnerz = centerpoint\_z + sizemax \* 0.5;

while (spawnerz < brainz - 1.23758547e+30) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

neuroblast();

} else {

glial\_cell\_2();

}

spawnerz = spawnerz + 6.187927353e+29;

}

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz - sizemax \* 0.012, (brainy - 7.217 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.867 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + sizemax \* 0.98;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.378) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.945) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.54) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz - sizemax \* 0.012, (brainy - 7.105 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + sizemax \* 0.98;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.378) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.945) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - 1.856378206e+29;

nerve\_junction\_right\_forth();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.44) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.485) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.124 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.14 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.377) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.74) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.4023 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.6249 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.73) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.5606 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.7942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.372) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

unused\_variable = 5;

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.805) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 5.9189 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.9501 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.105;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.105) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.801) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.9807 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.3537 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.08;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.08) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.795, (brainy - 6.55 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.95 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.12;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.7) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.05;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.05) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.79, (brainy - 6.5892 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.4981 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.125;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.69) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.05;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.03) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.05) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.785, (brainy - 6.0702 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.957 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.13;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.55) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.09;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.11) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.09) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.78, (brainy - 7.2511 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.8975 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.135;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.54) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.09;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.03) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.09) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.775, (brainy - 6.065 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.8 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.14;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.487) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.011;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.011) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.77, (brainy - 7.06 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.54 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.145;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.4835) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.011;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.345) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.011) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

unused\_variable = 5;

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.805) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.3316 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.9411 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.88 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.88) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.801) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.001 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.78 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.905 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.905) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.795, (brainy - 6.315 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.21 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.88;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.7) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.85 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.85) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.79, (brainy - 6.9426 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.4006 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.875;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.69) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.85 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.03) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.85) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.785, (brainy - 6.095 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.084 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.87;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.55) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.89 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.11) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.89) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.78, (brainy - 6.6439 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.8057 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.865;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.54) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.89 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.03) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.89) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.775, (brainy - 6.3226 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.301 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.86;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.487) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.801 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.345) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.801) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.77, (brainy - 6.8564 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.2589 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.855;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.484) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.801 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.801) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

unused\_variable = 5;

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.495) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.285 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.45 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.3131;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.37) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.3131) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.49) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.077 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.096 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.28;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.37) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.28) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.487, (brainy - 6.265 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.345 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.3;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.4) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.2;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.32) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.2) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.484, (brainy - 6.879 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.0579 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.31;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.39) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.2;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.2) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.481, (brainy - 6.043 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.9286 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.32;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.17) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.29;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.33) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.29) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

motor\_2b();

motor\_2c();

}

function motor\_2b() {

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.478, (brainy - 6.6639 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.3998 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.33;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.16) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.29;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.29) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.495) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.2895 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.4919 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.68 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.37) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.68) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.49) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.86 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.28 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.7131 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.37) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.7131) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.487, (brainy - 6.315 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.265 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.595;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.4) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.6 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.32) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.6) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.484, (brainy - 6.948 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.31 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.59;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.39) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.6 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.6) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.481, (brainy - 6.13 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.125 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.585;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.17) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.69 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.33) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.69) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + sizemax \* 0.478, (brainy - 6.3 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.22 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.58;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_back\_right();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.16) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.69 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.69) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

}

function motor\_2c() {

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.909) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.22 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.765 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (sizemin \* 0.656 - 1.856378206e+29);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.33) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.656) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.909) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.1 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.185 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.3441;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.33) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.336) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.9093) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.5093 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.86 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.35;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.299) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.4809) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.9103) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.481) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.9097) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.185 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.76 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.351;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.2995) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.4809) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.912) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.356) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.481) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.7175) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.573 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.6635 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.155) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

spawnerz2 = loc5[0];

spawnery2 = loc5[1];

spawnerx2 = loc5[2];

nerve\_trijunction\_right();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.355) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.245) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

loc5 = [spawnerz2, spawnery2, spawnerx2 + 1.856378206e+29];

loc6 = [spawnerz2 + 3.093963676e+28, spawnery2 - 3.093963676e+28, spawnerx2];

nerve\_junction\_left\_forth();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.245) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

}

function motor\_1() {

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.68) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 7.001 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.78 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.58) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.62) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.3316 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.9411 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.58) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.365) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.86 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.28 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.58) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.305) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.2895 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.4919 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.58) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.68) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.9807 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.3532 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.62) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 5.9189 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.9501 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.365) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 7.077 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.096 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.305) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.285 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.45 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.735) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.3) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

unused\_variable = 5;

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.685) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.9426 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.4006 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.88) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.625) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.315 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.21 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.88) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.36) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.948 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.31 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.88) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.31) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.315 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.265 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.88) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.67) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.6439 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.8057 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.92) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.617) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.095 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.084 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.92) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.35735) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.3 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.22 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.92) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.303) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.13 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.125 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.92) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.66) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.8564 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.2589 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.249) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.921) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.665) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.3226 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.301 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.21) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.921) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.345) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.943 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.2039 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.249) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.921) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.35) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 5.9393 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.1896 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.9891) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - cedex;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.21) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.921) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

unused\_variable = 5;

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz - sizemax \* 0.012, (brainy - 7.217 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.867 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + sizemax \* 0.98;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.378) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.945) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.54) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [brainz - sizemax \* 0.012, (brainy - 7.105 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + sizemax \* 0.98;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.378) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = centerpoint\_z + (Math.round((sizemax \* 0.945) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29);

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - 1.856378206e+29;

nerve\_junction\_right\_forth();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.44) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.485) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.124 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.14 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.377) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.74) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.4023 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.6249 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.379) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.73) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.5606 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.7942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.011;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.221;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_up();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (Math.floor((sizemin \* 0.372) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_left\_down();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

unused\_variable = 5;

motor\_1b();

motor\_1c();

if (unused\_variable == 5) {

spawnerx = centerpoint\_x - sizemid \* 0.5;

spawnery = brainy + 6.187927353e+29;

spawnerz = centerpoint\_z + sizemax \* 0.31;

while (spawnerz < brainz - 1.23758547e+30) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

neuroblast();

} else {

glial\_cell\_2();

}

spawnerz = spawnerz + 6.187927353e+29;

}

}

}

function motor\_1b() {

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.685) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.5892 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.4981 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.02) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.625) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.55 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.95 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.02) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.36) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.879 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.0579 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.02) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.31) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.265 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.345 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.725) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.02) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.67) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.2511 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.8975 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.617) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.0702 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.957 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.35735) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.6639 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.3998 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.303) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.043 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.9286 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.4) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.06) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.66) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.06 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.54 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.249) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.0101) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.665) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.065 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.8 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.21) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.0101) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.345) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.785 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.575 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.249) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.0101) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.35) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.1 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.55 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.01001) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + cedex;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.21) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.0101) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

}

function motor\_1c() {

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.28) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 5.9358 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.0272 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.79;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.785) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.55) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.28) / 1.23758547e+30) \* 1.23758547e+30 + (1.553169765e+29 + 1.856378206e+29)), (brainy - 6.1686 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.5574 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.79;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_back\_down();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.785) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.33) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.5855) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.573 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.6635 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.203;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.65) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.23) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.57) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.4023 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 6.6249 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.203;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.202) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.25) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.563) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 6.5606 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.7942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.203;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.2057) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.25) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.54) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.124 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.14 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - sizemin \* 0.7951;

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_right\_down();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.2001;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.2045) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.201) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.76) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.217 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.867 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (sizemin \* 0.77 + 1.856378206e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.91) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_down\_back();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.53) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_left\_up();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.79) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.76) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.105 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 4.942 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_up\_forth();

loc5[1] = loc6[1];

loc6[1] = centerpoint\_y - (sizemin \* 0.77 + 1.856378206e+30);

motor\_nerve\_y();

loc5[1] = loc6[1];

loc6[1] = loc5[1] - cedex;

nerve\_junction\_down\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.91) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_down\_back();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.53) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.204) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.765) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.22 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.765 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.451;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.96) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - sizemin \* 0.7105;

motor\_nerve\_x();

loc5[2] = loc6[2];

loc6[2] = loc5[2] - 1.856378206e+29;

nerve\_junction\_right\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.318) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_up\_left();

loc5[2] = loc6[2];

loc6[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.711) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

motor\_unit\_1();

}

if (unused\_variable == 5) {

cedex = 1.856378206e+29;

loc5 = [centerpoint\_z + (Math.round((sizemax \* 0.765) / 1.23758547e+30) \* 1.23758547e+30 + 1.553169765e+29), (brainy - 7.1 \* 3.093963676e+31) + 3.093963676e+28, (brainx - 5.185 \* 3.093963676e+31) + 3.093963676e+28];

loc6 = [brainz + 0, loc5[1] - 1.856378206e+29, loc5[2] - 1.856378206e+29];

motor\_nerve\_z();

loc6[0] = loc5[0];

loc5[0] = loc6[0] - cedex;

nerve\_junction\_down\_forth();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - sizemin \* 0.451;

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] + cedex;

nerve\_junction\_up\_forth();

loc5[0] = loc6[0];

loc6[0] = centerpoint\_z + (Math.round((sizemax \* 0.96) / 1.23758547e+30) \* 1.23758547e+30 - 3.032084403e+28);

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + cedex;

nerve\_junction\_back\_right();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - sizemin \* 0.289;

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_left\_down();

loc6[1] = loc5[1];

loc5[1] = centerpoint\_y - ((Math.floor((sizemin \* 0.318) / 2.475170941e+30) \* 2.475170941e+30 - 1.23758547e+30) + 1.856378206e+29);

motor\_nerve\_y();

loc6[1] = loc5[1];

loc5[1] = loc6[1] - cedex;

nerve\_junction\_right\_up();

loc6[2] = loc5[2];

loc5[2] = centerpoint\_x - (Math.ceil((sizemin \* 0.28) / 2.475170941e+30) \* 2.475170941e+30 + 1.23758547e+30);

motor\_nerve\_x();

loc6[2] = loc5[2];

loc5[2] = loc6[2] + 1.856378206e+29;

nerve\_junction\_forth\_left();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.05194765e+30;

motor\_nerve\_z();

loc5[0] = loc6[0];

loc6[0] = loc5[0] + 1.856378206e+29;

nerve\_junction\_back\_left();

motor\_unit\_1();

}

}

function nerve\_junction\_back\_down() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0];

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_x();

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 0;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

lipid2 = 5 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_back\_left() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0];

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_x();

lipid2 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

lipid1 = 5 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_back\_right() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0];

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

lipid2 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 0;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

lipid1 = 5 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 0;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_back\_up() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0];

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 5 \* 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_x();

lipid2 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 1.546981838e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_down\_back() {

nerve\_junction\_back\_down();

}

function nerve\_junction\_down\_forth() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_x();

lipid2 = 1 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_y();

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 0;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

lipid2 = 1 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_forth\_left() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid1 = 1 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_forth\_right() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_y();

lipid2 = 1 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_x();

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 0;

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_z();

lipid1 = 1 \* 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_left\_down() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 0;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 0;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_y();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_left\_forth() {

nerve\_junction\_forth\_left();

}

function nerve\_junction\_left\_up() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

cursor2y = loc5[1] - 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_right\_down() {

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 0;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_right\_forth() {

nerve\_junction\_forth\_right();

}

function nerve\_junction\_right\_up() {

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 0;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 5 \* 3.093963676e+28;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_up\_back() {

nerve\_junction\_back\_up();

}

function nerve\_junction\_up\_forth() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 1.546981838e+29;

lipid1 = 4 \* 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 5 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 1.546981838e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 1.546981838e+29;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_up\_left() {

nerve\_junction\_left\_up();

}

function nerve\_trijunction\_down() {

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 6 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 0;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_x();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

lipid\_wall\_y();

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

cursor2x = loc5.slice(-1)[0] - 0;

lipid1 = 6 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_trijunction\_right() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 0;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 4 \* 3.093963676e+28;

lipid2 = 6 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2x = loc5.slice(-1)[0] - 0;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid1 = 1 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 1.546981838e+29;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_y();

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

lipid\_wall\_x();

cursor2y = loc5[1] - 1.546981838e+29;

lipid\_wall\_x();

cursor2y = loc5[1] - 0;

cursor2x = loc5.slice(-1)[0] - 1.546981838e+29;

lipid1 = 6 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

nerve\_junction\_default\_fluids();

cells\_list[cells\_list.length - 1] = ['nerve junction', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]];

}

function nerve\_junction\_default\_fluids() {

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_cytosol();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function nerve\_z() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0];

lipid2 = loc6[0] - loc5[0];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = loc5[1] - 5 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor1y = loc5[1] - 1.1 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1z = loc5[0];

repeat\_endcell1 = Math.round((loc6[0] - loc5[0]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1z = cursor1z + 10 \* 3.093963676e+28;

}

cursor1z = loc5[0];

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1y = loc5[1] - 2.2 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc6[0] - loc5[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 14; countcell2++) {

for (countcell3 = 0; countcell3 < 20; countcell3++) {

synaptic\_vesicle\_1();

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = cursor1x + 4.950341882e+27 \* 20;

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* 14;

cursor1z = cursor1z + 4.950341882e+27;

}

default\_nerve\_z();

}

function motor\_nerve\_z() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0];

lipid2 = loc6[0] - loc5[0];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = loc5[1] - 5 \* 3.093963676e+28;

lipid\_wall\_y();

cursor2y = loc5[1] - 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 5 \* 3.093963676e+28;

lipid\_wall\_x();

cursor1y = loc5[1] - 1.1 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1z = loc5[0];

repeat\_endcell1 = Math.round((loc6[0] - loc5[0]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1z = cursor1z + 10 \* 3.093963676e+28;

}

cursor1z = loc5[0];

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1y = loc5[1] - 2.2 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc6[0] - loc5[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 14; countcell2++) {

for (countcell3 = 0; countcell3 < 20; countcell3++) {

synaptic\_vesicle\_2();

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = cursor1x + 4.950341882e+27 \* 20;

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* 14;

cursor1z = cursor1z + 4.950341882e+27;

}

default\_nerve\_z();

}

function default\_nerve\_z() {

cursor1y = loc5[1] - 3.093963676e+28;

cursor1z = loc5[0] + 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 4.331549147e+26 \* 1;

density\_parameter\_y = 4.331549147e+26 \* 1;

density\_parameter\_z = 4.331549147e+26;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.537050215e+26 \* 1;

density\_parameter\_y = 2.537050215e+26 \* 1;

density\_parameter\_z = 2.537050215e+26;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

heme();

} else {

peptidesequence = '1 merpepelir qswravsrsp lehgtvlfar lfalepdllp lfqyncrqfs spedclsspe 61 fldhirkvml vidaavtnve dlssleeyla slgrkhravg vklssfstvg esllymlekc 121 lgpaftpatr aawsqlygav vqamsrgwdg e';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.05194765e+27 \* 1.13;

density\_parameter\_y = 1.04194765e+27 \* 0.97;

density\_parameter\_z = 1.06194765e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.803131091e+27 \* 1.79;

density\_parameter\_y = 2.803131091e+27 \* 0.83;

density\_parameter\_z = 2.803131091e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.666996689e+27 \* 1;

density\_parameter\_y = 2.666996689e+27 \* 1;

density\_parameter\_z = 2.666996689e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] - 0;

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 10);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0];

cursor1y = loc5[1] - 5 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 10);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((loc6[0] - loc5[0]) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if ((cursor1z > loc5[0] + 3.093963676e+28 && cursor1z < loc5[1] + 5 \* 3.093963676e+28) && (cursor1y < loc5[1] - 3.093963676e+28 && cursor1y > loc5[1] - 1.546981838e+26)) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cells\_list.push(['nerve', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]]);

}

function nerve\_y() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1];

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[1] - loc6[1];

lipid\_wall\_z();

lipid2 = loc5[1] - loc6[1];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2z = loc5[0] + 5 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[1] - loc6[1];

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 5 \* 3.093963676e+28;

lipid2 = loc5[1] - loc6[1];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5[1] - loc6[1]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1y = cursor1y - 10 \* 3.093963676e+28;

}

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0] - 2.2 \* 3.093963676e+28;

cursor1y = loc5[1];

repeat\_endcell1 = Math.round((loc5[0] - loc6[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 20; countcell2++) {

for (countcell3 = 0; countcell3 < 14; countcell3++) {

synaptic\_vesicle\_1();

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = cursor1x + 4.950341882e+27 \* 14;

cursor1z = cursor1y + 4.950341882e+27;

}

cursor1z = cursor1y - 4.950341882e+27 \* 20;

cursor1y = cursor1y - 4.950341882e+27;

}

default\_nerve\_y();

}

function motor\_nerve\_y() {

cursor2x = loc5.slice(-1)[0] - 3.093963676e+28;

cursor2y = loc5[1];

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[1] - loc6[1];

lipid\_wall\_z();

lipid2 = loc5[1] - loc6[1];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

cursor2z = loc5[0] + 5 \* 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[1] - loc6[1];

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor2x = loc5.slice(-1)[0] - 5 \* 3.093963676e+28;

lipid2 = loc5[1] - loc6[1];

lipid1 = 4 \* 3.093963676e+28;

lipid\_wall\_x();

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 1.1 \* 3.093963676e+28;

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5[1] - loc6[1]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1y = cursor1y - 10 \* 3.093963676e+28;

}

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0] - 2.2 \* 3.093963676e+28;

cursor1y = loc5[1];

repeat\_endcell1 = Math.round((loc5[0] - loc6[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 20; countcell2++) {

for (countcell3 = 0; countcell3 < 14; countcell3++) {

synaptic\_vesicle\_2();

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = cursor1x + 4.950341882e+27 \* 14;

cursor1z = cursor1y + 4.950341882e+27;

}

cursor1z = cursor1y - 4.950341882e+27 \* 20;

cursor1y = cursor1y - 4.950341882e+27;

}

default\_nerve\_y();

}

function default\_nerve\_y() {

cursor1y = loc5[1];

cursor1z = loc5[0] + 3.093963676e+28;

cursor1x = loc5.slice(-1)[0] - 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 4.331549147e+26 \* 1;

density\_parameter\_y = 4.331549147e+26 \* 1;

density\_parameter\_z = 4.331549147e+26;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 2.537050215e+26 \* 1;

density\_parameter\_y = 2.537050215e+26 \* 1;

density\_parameter\_z = 2.537050215e+26;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

heme();

} else {

peptidesequence = '1 merpepelir qswravsrsp lehgtvlfar lfalepdllp lfqyncrqfs spedclsspe 61 fldhirkvml vidaavtnve dlssleeyla slgrkhravg vklssfstvg esllymlekc 121 lgpaftpatr aawsqlygav vqamsrgwdg e';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 1.05194765e+27 \* 1.13;

density\_parameter\_y = 1.04194765e+27 \* 0.97;

density\_parameter\_z = 1.06194765e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 2.803131091e+27 \* 1.79;

density\_parameter\_y = 2.803131091e+27 \* 0.83;

density\_parameter\_z = 2.803131091e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 2.666996689e+27 \* 1;

density\_parameter\_y = 2.666996689e+27 \* 1;

density\_parameter\_z = 2.666996689e+27;

generation\_parameter\_x = Math.round((4 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] - 0;

cursor1y = loc5[1];

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 5 \* 3.093963676e+28;

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((6 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((loc5[1] - loc6[1]) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if ((cursor1z > loc5[0] + 3.093963676e+28 && cursor1z < loc5[1] + 5 \* 3.093963676e+28) && (cursor1x < loc5[2] - 3.093963676e+28 && cursor1x > loc5[2] - 1.546981838e+26)) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cells\_list.push(['nerve', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]]);

}

function nerve\_x() {

cursor2x = loc5.slice(-1)[0];

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[2] - loc6[2];

lipid\_wall\_z();

lipid\_wall\_y();

cursor2z = loc5[0] + 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 5 \* 3.093963676e+28;

lipid\_wall\_y();

cursor1y = loc5[1] - 1.1 \* 3.093963676e+28;

cursor1x = loc5[2];

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5[2] - loc6[2]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1x = cursor1x - 10 \* 3.093963676e+28;

}

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

cursor1x = loc5[2];

cursor1y = loc5[1] - 2.2 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 20; countcell2++) {

for (countcell3 = 0; countcell3 < 14; countcell3++) {

synaptic\_vesicle\_1();

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* 14;

cursor1z = cursor1y + 4.950341882e+27;

}

cursor1z = cursor1y - 4.950341882e+27 \* 20;

cursor1x = cursor1x - 4.950341882e+27;

}

default\_nerve\_x();

}

function motor\_nerve\_x() {

cursor2x = loc5.slice(-1)[0];

cursor2y = loc5[1] - 3.093963676e+28;

cursor2z = loc5[0] + 3.093963676e+28;

lipid2 = 4 \* 3.093963676e+28;

lipid1 = loc5[2] - loc6[2];

lipid\_wall\_z();

lipid\_wall\_y();

cursor2z = loc5[0] + 5 \* 3.093963676e+28;

lipid\_wall\_z();

cursor2z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 5 \* 3.093963676e+28;

lipid\_wall\_y();

cursor1y = loc5[1] - 1.1 \* 3.093963676e+28;

cursor1x = loc5[2];

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5[2] - loc6[2]) / (10 \* 3.093963676e+28) - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

mitochondrion();

cursor1x = cursor1x - 10 \* 3.093963676e+28;

}

cursor1z = loc5[0] + 1.1 \* 3.093963676e+28;

cursor1x = loc5[2];

cursor1y = loc5[1] - 2.2 \* 3.093963676e+28;

repeat\_endcell1 = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

for (countcell2 = 0; countcell2 < 20; countcell2++) {

for (countcell3 = 0; countcell3 < 14; countcell3++) {

synaptic\_vesicle\_2();

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* 14;

cursor1z = cursor1y + 4.950341882e+27;

}

cursor1z = cursor1y - 4.950341882e+27 \* 20;

cursor1x = cursor1x - 4.950341882e+27;

}

default\_nerve\_x();

}

function default\_nerve\_x() {

cursor1y = loc5[1] - 3.093963676e+28;

cursor1z = loc5[0] + 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 4.331549147e+26 \* 1;

density\_parameter\_y = 4.331549147e+26 \* 1;

density\_parameter\_z = 4.331549147e+26;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.537050215e+26 \* 1;

density\_parameter\_y = 2.537050215e+26 \* 1;

density\_parameter\_z = 2.537050215e+26;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (unused\_variable == 5) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

heme();

} else {

peptidesequence = '1 merpepelir qswravsrsp lehgtvlfar lfalepdllp lfqyncrqfs spedclsspe 61 fldhirkvml vidaavtnve dlssleeyla slgrkhravg vklssfstvg esllymlekc 121 lgpaftpatr aawsqlygav vqamsrgwdg e';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.05194765e+27 \* 1.13;

density\_parameter\_y = 1.04194765e+27 \* 0.97;

density\_parameter\_z = 1.06194765e+27;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.803131091e+27 \* 1.79;

density\_parameter\_y = 2.803131091e+27 \* 0.83;

density\_parameter\_z = 2.803131091e+27;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] + 3.093963676e+28;

cursor1y = loc5[1] - 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 2.666996689e+27 \* 1;

density\_parameter\_y = 2.666996689e+27 \* 1;

density\_parameter\_z = 2.666996689e+27;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((4 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((4 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0] - 0;

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 10);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1z = loc5[0];

cursor1y = loc5[1] - 5 \* 3.093963676e+28;

cursor1x = loc5.slice(-1)[0];

if (unused\_variable == 5) {

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((loc5[2] - loc6[2]) / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 10);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 390);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masqkrpsqr hgskylatas tmdharhgfl prhrdtgild sigrffggdr gapkrgsgkv 61 pwlkpgrspl psharsqpgl cnmykdshhp artahygslp qkshgrtqde npvvhffkni 121 vtprtpppsq gkgrglslsr fswgaegqrp gfgyggrasd yksahkgfkg vdaqgtlski 181 fklggrdsrs gspmarr';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round((loc5.slice(-1)[0] - loc6.slice(-1)[0]) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6 \* 3.093963676e+28) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if ((cursor1z > loc5[0] + 3.093963676e+28 && cursor1z < loc5[1] + 5 \* 3.093963676e+28) && (cursor1y < loc5[1] - 3.093963676e+28 && cursor1y > loc5[1] - 1.546981838e+26)) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cells\_list.push(['motor nerve', loc5[0], loc5[1], loc5.slice(-1)[0], loc6[0], loc6[1], loc6.slice(-1)[0]]);

}

function randomize\_sarcomere\_protein() {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 94) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 90) {

if (randomizer <= 90) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 65) {

peptidesequence = ' 1 mapkkdvkkp vaaaaaapap apapapapap akpkeekidl saikiefske qqdefkeafl 61 lfdrtgdski tlsqvgdvlr algtnptnae vrkvlgnpsn eelnakkief eqflpmmqai 121 snnkdqatye dfveglrvfd kegngtvmga elrhvlatlg ekmkeeevea lmagqedsng 181 cinyeafvkh imsi';

} else if (randomizer <= 130) {

peptidesequence = ' 1 mapkrakrrt veggsssvfs mfdqtqiqef keaftvidqn rdgiidkedl rdtfaamgrl 61 nvkneeldam mkeasgpinf tvfltmfgek lkgadpedvi tgafkvldpe gkgtikkkfl 121 eellttqcdr fsqeeiknmw aafppdvggn vdyknicyvi thgdakdqe';

} else if (randomizer <= 150) {

peptidesequence = ' 10 20 30 40 50 MSSDSELAVF GEAAPFLRKS ERERIEAQNR PFDAKTSVFV AEPKESFVKG 60 70 80 90 100 TIQSREGGKV TVKTEGGATL TVKDDQVFPM NPPKYDKIED MAMMTHLHEP 110 120 130 140 150 AVLYNLKERY AAWMIYTYSG LFCVTVNPYK WLPVYKPEVV TAYRGKKRQE 160 170 180 190 200 APPHIFSISD NAYQFMLTDR ENQSILITGE SGAGKTVNTK RVIQYFATIA 210 220 230 240 250 VTGEKKKEEI TSGKIQGTLE DQIISANPLL EAFGNAKTVR NDNSSRFGKF 260 270 280 290 300 IRIHFGTTGK LASADIETYL LEKSRVVFQL KAERSYHIFY QITSNKKPEL 310 320 330 340 350 IEMLLITTNP YDYPFVSQGE ISVASIDDQE ELMATDSAID ILGFTNEEKV 360 370 380 390 400 SIYKLTGAVM HYGNLKFKQK QREEQAEPDG TEVADKAAYL QSLNSADLLK 410 420 430 440 450 ALCYPRVKVG NEYVTKGQTV EQVSNAVGAL AKAVYEKMFL WMVARINQQL 460 470 480 490 500 DTKQPRQYFI GVLDIAGFEI FDFNSLEQLC INFTNEKLQQ FFNHHMFVLE 510 520 530 540 550 QEEYKKEGIE WTFIDFGMDL AACIELIEKP MGIFSILEEE CMFPKATDTS 560 570 580 590 600 FKNKLYDQHL GKSANFQKPK VVKGKAEAHF ALIHYAGVVD YNITGWLEKN 610 620 630 640 650 KDPLNETVVG LYQKSAMKTL AQLFSGAQTA EGEGAGGGAK KGGKKKGSSF 660 670 680 690 700 QTVSALFREN LNKLMTNLRS THPHFVRCII PNETKTPGAM EHELVLHQLR 710 720 730 740 750 CNGVLEGIRI CRKGFPSRIL YADFKQRYKV LNASAIPEGQ FIDSKKASEK 760 770 780 790 800 LLASIDIDHT QYKFGHTKVF FKAGLLGLLE EMRDDKLAQL ITRTQARCRG 810 820 830 840 850 FLARVEYQRM VERREAIFCI QYNIRSFMNV KHWPWMKLFF KIKPLLKSAE 860 870 880 890 900 TEKEMATMKE EFQKIKDELA KSEAKRKELE EKMVTLLKEK NDLQLQVQAE 910 920 930 940 950 AEGLADAEER CDQLIKTKIQ LEAKIKEVTE RAEDEEEINA ELTAKKRKLE 960 970 980 990 1000 DECSELKKDI DDLELTLAKV EKEKHATENK VKNLTEEMAG LDETIAKLTK 1010 1020 1030 1040 1050 EKKALQEAHQ QTLDDLQAEE DKVNTLTKAK IKLEQQVDDL EGSLEQEKKL 1060 1070 1080 1090 1100 RMDLERAKRK LEGDLKLAQE SIMDIENEKQ QLDEKLKKKE FEISNLQSKI 1110 1120 1130 1140 1150 EDEQALGIQL QKKIKELQAR IEELEEEIEA ERASRAKAEK QRSDLSRELE 1160 1170 1180 1190 1200 EISERLEEAG GATSAQIEMN KKREAEFQKM RRDLEEATLQ HEATAATLRK 1210 1220 1230 1240 1250 KHADSVAELG EQIDNLQRVK QKLEKEKSEM KMEIDDLASN VETVSKAKGN 1260 1270 1280 1290 1300 LEKMCRTLED QLSELKSKEE EQQRLINDLT AQRGRLQTES GEFSRQLDEK 1310 1320 1330 1340 1350 EALVSQLSRG KQAFTQQIEE LKRQLEEEIK AKNALAHALQ SSRHDCDLLR 1360 1370 1380 1390 1400 EQYEEEQESK AELQRALSKA NTEVAQWRTK YETDAIQRTE ELEEAKKKLA 1410 1420 1430 1440 1450 QRLQAAEEHV EAVNAKCASL EKTKQRLQNE VEDLMLDVER TNAACAALDK 1460 1470 1480 1490 1500 KQRNFDKILA EWKQKCEETH AELEASQKEA RSLGTELFKI KNAYEESLDQ 1510 1520 1530 1540 1550 LETLKRENKN LQQEISDLTE QIAEGGKRIH ELEKIKKQVE QEKCELQAAL 1560 1570 1580 1590 1600 EEAEASLEHE EGKILRIQLE LNQVKSEVDR KIAEKDEEID QLKRNHIRIV 1610 1620 1630 1640 1650 ESMQSTLDAE IRSRNDAIRL KKKMEGDLNE MEIQLNHANR MAAEALRNYR 1660 1670 1680 1690 1700 NTQGILKDTQ IHLDDALRSQ EDLKEQLAMV ERRANLLQAE IEELRATLEQ 1710 1720 1730 1740 1750 TERSRKIAEQ ELLDASERVQ LLHTQNTSLI NTKKKLETDI SQMQGEMEDI 1760 1770 1780 1790 1800 LQEARNAEEK AKKAITDAAM MAEELKKEQD TSAHLERMKK NMEQTVKDLQ 1810 1820 1830 1840 1850 LRLDEAEQLA LKGGKKQIQK LEARVRELEG EVESEQKRNA EAVKGLRKHE 1860 1870 1880 1890 1900 RRVKELTYQT EEDRKNILRL QDLVDKLQAK VKSYKRQAEE AEEQSNTNLA 1910 1920 1930 1940 KFRKLQHELE EAEERADIAE SQVNKLRVKS REVHTKVISE E ';

}

} else if (randomizer <= 100) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 25) {

peptidesequence = ' 10 20 30 40 50 MDAIKKKMQM LKLDKENALD RAEQAEADKK AAEDRSKQLE DELVSLQKKL 60 70 80 90 100 KGTEDELDKY SEALKDAQEK LELAEKKATD AEADVASLNR RIQLVEEELD 110 120 130 140 150 RAQERLATAL QKLEEAEKAA DESERGMKVI ESRAQKDEEK MEIQEIQLKE 160 170 180 190 200 AKHIAEDADR KYEEVARKLV IIESDLERAE ERAELSEGKC AELEEELKTV 210 220 230 240 250 TNNLKSLEAQ AEKYSQKEDR YEEEIKVLSD KLKEAETRAE FAERSVTKLE 260 270 280 KSIDDLEDEL YAQKLKYKAI SEELDHALND MTSI ';

} else if (randomizer <= 50) {

peptidesequence = ' 10 20 30 40 50 MDAIKKKMQM LKLDKENAID RAEQAEADKK QAEDRCKQLE EEQQALQKKL 60 70 80 90 100 KGTEDEVEKY SESVKEAQEK LEQAEKKATD AEADVASLNR RIQLVEEELD 110 120 130 140 150 RAQERLATAL QKLEEAEKAA DESERGMKVI ENRAMKDEEK MELQEMQLKE 160 170 180 190 200 AKHIAEDSDR KYEEVARKLV ILEGELERSE ERAEVAESKC GDLEEELKIV 210 220 230 240 250 TNNLKSLEAQ ADKYSTKEDK YEEEIKLLEE KLKEAETRAE FAERSVAKLE 260 270 280 KTIDDLEDEV YAQKMKYKAI SEELDNALND ITSL ';

} else if (randomizer <= 150) {

peptidesequence = '1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

}

}

} else if (randomizer <= 100) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 mslpfyqrch qhydlsyrnk dvrstvshyq rekkrsavyt qgstayssrs saahrresea 61 frrasasssq qqasqhalss evsrkaasay dygsshgltd sslllddyss klspkpkrak 121 hsllsgeeke nlpsdymvpi fsgrqkhvsg itdteeerik eaaayiaqrn llaseegitt 181 skqstaskqt taskqstask qstaskqsta srqstasrqs vvskqatsal qqeetsekks 241 rkvvirekae rlslrktlee tetyhaklne dhllhapefi ikprshtvwe kenvklhcsi 301 agwpeprvtw yknqvpinvh anpgkyiies rygmhtlein gcdfedtaqy rasamnvkge 361 lsayasvvvk rykgefdetr fhagastmpl sfgvtpygya srfeihfddk fdvsfgrege 421 tmslgcrvvi tpeikhfqpe iqwyrngvpl spskwvqtlw sgeratltfs hlnkedegly 481 tirvrmgeyy eqysayvfvr dadaeiegap aapldvkcle ankdyiiisw kqpavdggsp 541 ilgyfidkce vgtdswsqcn dtpvkfarfp vtgliegrsy ifrvravnkm gigfpsrvse 601 pvaaldpaek arlksrpsap wtgqiivtee epsegivpgp ptdlsvteat rsyvvlswkp 661 pgqrghegim yfvekceagt enwqrvntel pvksprfalf dlaegksycf rvrcsnsagv 721 gepseatevt vvgdkldipk apgkiipsrn tdtsvvvswe eskdakelvg yyieasvags 781 gkwepcnnnp vkgsrftchg lvtgqsyifr vravnaagls eysqdseaie vkaaigggvs 841 pdvcpalsde pggltasrgr vheaspptfq kdallgskpn kpslpsssqn lgqtevskvs 901 etvqeeltpp pqkaapqgks ksdplkkktd rappsppcdi tclesfrdsm vlgwkqpdki 961 ggaeitgyyv nyrevidgvp gkwreanvka vseeaykisn lkenmvyqfq vaamnmaglg 1021 apsavsecfk ceewtiavpg pphslkcsev rkdslvlqwk ppvhsgrtpv tgyfvdlkea 1081 kakedqwrgl neaaiknvyl kvrglkegvs yvfrvrainq agvgkpsdla gpvvaetrpg 1141 tkevvvnvdd dgvislnfec dkmtpksefs wskdyvsted sprleveskg nktkmtfkdl 1201 gmddlgiysc dvtdtdgias sylideeelk rllalshehk fptvpvksel aveilekgqv 1261 rfwmqaekls gnakvnyifn ekeifegpky kmhidrntgi iemfmeklqd edegtytfql 1321 qdgkatnhst vvlvgdvfkk lqkeaefqrq ewirkqgphf veylswevtg ecnvllkckv 1381 anikkethiv wykdereisv dekhdfkdgi ctllitefsk kdagiyevil kddrgkdksr 1441 lklvdeafke lmmevckkia lsatdlkiqs taegiqlysf vtyyvedlkv nwshngsair 1501 ysdrvktgvt geqiwlqine ptpndkgkyv melfdgktgh qktvdlsgqa ydeayaefqr 1561 lkqaaiaekn rarvlgglpd vvtiqegkal nltcnvwgdp ppevswlkne kalasddhcn 1621 lkfeagrtay ftingvstad sgkyglvvkn kygsetsdft vsvfipeeea rmaaleslkg 1681 gkkak';

} else if (randomizer <= 100) {

peptidesequence = '1 mpepgkkpvs afskkprsve vaagspavfe aeteragvkv rwqrggsdis asnkyglate 61 gtrhtltvre vgpadqgsya viagsskvkf dlkvieaeka epmlapapap aeatgapgea 121 papaaelges apspkgsssa alngptpgap ddpiglfvmr pqdgevtvgg sitfsarvag 181 asllkppvvk wfkgkwvdls skvgqhlqlh dsydraskvy lfelhitdaq paftgsyrce 241 vstkdkfdcs nfnltvheam gtgdldllsa frrtslaggg rrisdshedt gildfssllk 301 krdsfrtprd skleapaeed vweilrqapp seyeriafqy gvtdlrgmlk rlkgmrrdek 361 kstafqkkle payqvskghk irltveladh daevkwlkng qeiqmsgsky ifesigakrt 421 ltisqcslad daayqcvvgg ekcstelfvk eppvlitrpl edqlvmvgqr vefecevsee 481 gaqvkwlkdg veltreetfk yrfkkdgqrh hliineamle daghyalcts ggqalaeliv 541 qekklevyqs iadlmvgakd qavfkcevsd envrgvwlkn gkelvpdsri kvshigrvhk 601 ltiddvtpad eadysfvpeg facnlsaklh fmevkidfvp rqeppkihld cpgripdtiv 661 vvagnklrld vpisgdpapt viwqkaitqg nkaparpapd apedtgdsde wvfdkkllce 721 tegrvrvett kdrsiftveg aekedegvyt vtvknpvged qvnltvkvid vpdapaapki 781 snvgedsctv qweppaydgg qpilgyiler kkkksyrwmr lnfdliqels hearrmiegv 841 vyemrvyavn aigmsrpspa sqpfmpigpp septhlaved vsdttvslkw rppervgagg 901 ldgysveycp egcsewvaal qgltehtsil vkdlptgarl lfrvrahnma gpgapvttte 961 pvtvqeilqr prlqlprhlr qtiqkkvgep vnllipfqgk prpqvtwtke gqplageevs 1021 irnsptdtil firaarrvhs gtyqvtvrie nmedkatlvl qvvdkpsppq dlrvtdawgl 1081 nvalewkppq dvgntelwgy tvqkadkktm ewftvlehyr rthcvvpeli igngyyfrvf 1141 sqnmvgfsdr aattkepvfi prpgityepp nykaldfsea psftqplvnr sviagytaml 1201 ccavrgspkp kiswfkngld lgedarfrmf skqgvltlei rkpcpfdggi yvcratnlqg 1261 earcecrlev rvpq';

}

}

} else if (randomizer <= 100) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = ' 1 msdteeqeye eeqpeeeaae eeeeapeepe pvaepeeerp kpsrpvvppl ippkipeger 61 vdfddihrkr mekdllelqt lidvhfeqrk keeeelvalk erierrrser aeqqrfrtek 121 ererqaklae ekmrkeeeea kkraeddakk kkvlsnmgah fggylvkaeq krgkrqtgre 181 mkvrilserk kpldidymge eqlrarsawl ppsqpscpar ekaqelsdwi hqlesekfdl 241 maklkqqkye invlynrish aqkfrkgagk grvggrwk';

} else if (randomizer <= 70) {

peptidesequence = '1 mtdqqaears ylseemiaef kaafdmfdad gggdisvkel gtvmrmlgqt ptkeeldaii 61 eevdedgsgt idfeeflvmm vrqmkedakg kseeelaecf rifdrnadgy idpeelaeif 121 rasgehvtde eieslmkdgd knndgridfd eflkmmegvq';

} else if (randomizer <= 100) {

peptidesequence = ' 1 mpeverkpki tasrklllks lmlakakecw eqeheereae kvrylaerip tlqtrglsls 61 alqdlcrelh akvevvdeer ydieakclhn treikdlklk vmdlrgkfkr pplrrvrvsa 121 damlrallgs khkvsmdlra nlksvkkedt ekerpvevgd wrknveamsg megrkkmfda 181 aksptsq';

}

}

} else if (randomizer <= 100) {

peptidesequence = '1 mlwweevedc yeredvqkkt ftkwvnaqfs kfgkqhienl fsdlqdgrrl ldllegltgq 61 klpkekgstr vhalnnvnka lrvlqnnnvd lvnigstdiv dgnhkltlgl iwniilhwqv 121 knvmknimag lqqtnsekil lswvrqstrn ypqvnvinft tswsdglaln alihshrpdl 181 fdwnsvvcqq satqrlehaf niaryqlgie klldpedvdt typdkksilm yitslfqvlp 241 qqvsieaiqe vemlprppkv tkeehfqlhh qmhysqqitv slaqgyerts spkprfksya 301 ytqaayvtts dptrspfpsq hleapedksf gsslmesevn ldryqtalee vlswllsaed 361 tlqaqgeisn dvevvkdqfh thegymmdlt ahqgrvgnil qlgskligtg klsedeetev 421 qeqmnllnsr weclrvasme kqsnlhrvlm dlqnqklkel ndwltkteer trkmeeeplg 481 pdledlkrqv qqhkvlqedl eqeqvrvnsl thmvvvvdes sgdhataale eqlkvlgdrw 541 anicrwtedr wvllqdillk wqrlteeqcl fsawlseked avnkihttgf kdqnemlssl 601 qklavlkadl ekkkqsmgkl yslkqdllst lknksvtqkt eawldnfarc wdnlvqklek 661 staqisqavt ttqpsltqtt vmetvttvtt reqilvkhaq eelpppppqk krqitvdsei 721 rkrldvdite lhswitrsea vlqspefaif rkegnfsdlk ekvnaierek aekfrklqda 781 srsaqalveq mvnegvnads ikqaseqlns rwiefcqlls erlnwleyqn niiafynqlq 841 qleqmtttae nwlkiqpttp septaiksql kickdevnrl sdlqpqierl kiqsialkek 901 gqgpmfldad fvaftnhfkq vfsdvqarek elqtifdtlp pmryqetmsa irtwvqqset 961 klsipqlsvt dyeimeqrlg elqalqsslq eqqsglyyls ttvkemskka pseisrkyqs 1021 efeeiegrwk klssqlvehc qkleeqmnkl rkiqnhiqtl kkwmaevdvf lkeewpalgd 1081 seilkkqlkq crllvsdiqt iqpslnsvne ggqkikneae pefasrlete lkelntqwdh 1141 mcqqvyarke alkgglektv slqkdlsemh ewmtqaeeey lerdfeyktp delqkaveem 1201 krakeeaqqk eakvklltes vnsviaqapp vaqealkkel etlttnyqwl ctrlngkckt 1261 leevwacwhe llsylekank wlnevefklk ttenipggae eisevldsle nlmrhsednp 1321 nqirilaqtl tdggvmdeli neeletfnsr wrelheeavr rqklleqsiq saqetekslh 1381 liqesltfid kqlaayiadk vdaaqmpqea qkiqsdltsh eisleemkkh nqgkeaaqrv 1441 lsqidvaqkk lqdvsmkfrl fqkpanfeqr lqeskmilde vkmhlpalet ksveqevvqs 1501 qlnhcvnlyk slsevkseve mviktgrqiv qkkqtenpke ldervtalkl hynelgakvt 1561 erkqqlekcl klsrkmrkem nvltewlaat dmeltkrsav egmpsnldse vawgkatqke 1621 iekqkvhlks itevgealkt vlgkketlve dklsllnsnw iavtsraeew lnllleyqkh 1681 metfdqnvdh itkwiiqadt lldesekkkp qqkedvlkrl kaelndirpk vdstrdqaan 1741 lmanrgdhcr klvepqisel nhrfaaishr iktgkasipl keleqfnsdi qkllepleae 1801 iqqgvnlkee dfnkdmnedn egtvkellqr gdnlqqritd erkreeikik qqllqtkhna 1861 lkdlrsqrrk kaleishqwy qykrqaddll kclddiekkl aslpeprder kikeidrelq 1921 kkkeelnavr rqaeglsedg aamaveptqi qlskrwreie skfaqfrrln faqihtvree 1981 tmmvmtedmp leisyvpsty lteithvsqa lleveqllna pdlcakdfed lfkqeeslkn 2041 ikdslqqssg ridiihskkt aalqsatpve rvklqealsq ldfqwekvnk mykdrqgrfd 2101 rsvekwrrfh ydikifnqwl teaeqflrkt qipenwehak ykwylkelqd gigqrqtvvr 2161 tlnatgeeii qqssktdasi lqeklgslnl rwqevckqls drkkrleeqk nilsefqrdl 2221 nefvlwleea dniasiplep gkeqqlkekl eqvkllveel plrqgilkql netggpvlvs 2281 apispeeqdk lenklkqtnl qwikvsralp ekqgeieaqi kdlgqlekkl edleeqlnhl 2341 llwlspirnq leiynqpnqe gpfdvketei avqakqpdve eilskgqhly kekpatqpvk 2401 rkledlssew kavnrllqel rakqpdlapg lttigasptq tvtlvtqpvv tketaiskle 2461 mpsslmlevp aladfnrawt eltdwlslld qviksqrvmv gdledinemi ikqkatmqdl 2521 eqrrpqleel itaaqnlknk tsnqeartii tdrieriqnq wdevqehlqn rrqqlnemlk 2581 dstqwleake eaeqvlgqar akleswkegp ytvdaiqkki tetkqlakdl rqwqtnvdva 2641 ndlalkllrd ysaddtrkvh miteninasw rsihkrvser eaaleethrl lqqfpldlek 2701 flawlteaet tanvlqdatr kerlledskg vkelmkqwqd lqgeieahtd vyhnldensq 2761 kilrslegsd davllqrrld nmnfkwselr kkslnirshl eassdqwkrl hlslqellvw 2821 lqlkddelsr qapiggdfpa vqkqndvhra fkrelktkep vimstletvr iflteqpleg 2881 leklyqepre lppeeraqnv trllrkqaee vnteweklnl hsadwqrkid etlerlrelq 2941 eatdeldlkl rqaevikgsw qpvgdllids lqdhlekvka lrgeiaplke nvshvndlar 3001 qlttlgiqls pynlstledl ntrwkllqva vedrvrqlhe ahrdfgpasq hflstsvqgp 3061 weraispnkv pyyinhetqt tcwdhpkmte lyqsladlnn vrfsayrtam klrrlqkalc 3121 ldllslsaac daldqhnlkq ndqpmdilqi inclttiydr leqehnnlvn vplcvdmcln 3181 wllnvydtgr tgrirvlsfk tgiislckah ledkyrylfk qvasstgfcd qrrlglllhd 3241 siqiprqlge vasfggsnie psvrscfqfa nnkpeieaal fldwmrlepq smvwlpvlhr 3301 vaaaetakhq akcnickecp iigfryrslk hfnydicqsc ffsgrvakgh kmhypmveyc 3361 tpttsgedvr dfakvlknkf rtkryfakhp rmgylpvqtv legdnmetpv tlinfwpvds 3421 apasspqlsh ddthsriehy asrlaemens ngsylndsis pnesiddehl liqhycqsln 3481 qdsplsqprs paqilisles eergeleril adleeenrnl qaeydrlkqq hehkglsplp 3541 sppemmptsp qsprdaelia eakllrqhkg rlearmqile dhnkqlesql hrlrqlleqp 3601 qaeakvngtt vsspstslqr sdssqpmllr vvgsqtsdsm geedllsppq dtstgleevm 3661 eqlnnsfpss rgrntpgkpm redtm';

}

protein();

}

}

function myocyte() {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+30 - (4.085906752e+25 + 1);

lipid1 = 1.23758547e+30 - 1.23758547e+27;

cursor2z = spawnerz + 1 \* 1;

cursor2y = spawnery - 1 \* 6.187927353e+26;

cursor2x = spawnerx - 1 \* 6.187927353e+26;

lipid\_wall\_y();

lipid1 = 1.175706197e+30 - 1.23758547e+27;

lipid\_wall\_x();

cursor2y = spawnery - (1.23758547e+30 - 6.187927353e+26);

lipid\_wall\_x();

cursor2y = spawnery - 1 \* 6.187927353e+26;

cursor1y = spawnery - 2 \* 6.187927353e+26;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

cursor1z = spawnerz + 1 \* 1;

for (countcell1 = 0; countcell1 < 7000; countcell1++) {

set\_earth\_rotation();

cursor1x = cursor1x - 1 \* 1.23758547e+29;

peptidesequence = '1 mellspplrd vdltapdgsl csfattddfy ddpcfdspdl rffedldprl mhvgallkpe 61 ehshfpaavh papgaredeh vrapsghhqa grcllwacka ckrkttnadr rkaatmrerr 121 rlskvneafe tlkrctssnp nqrlpkveil rnairyiegl qallrdqdaa ppgaaaafya 181 pgplppgrgg ehysgdsdas sprsncsdgm mdysgppsga rrrncyegay yneapseprp 241 gksaavssld clssiveris tespaapall ladvpsespp rrqeaaapse gessgdptqs 301 pdaapqcpag anpnpiyqvl';

protein();

}

cursor1y = spawnerx - 3 \* 6.187927353e+26;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

nucleoid\_size\_y = 5.878530986e+28;

nucleoid\_size\_z = 1.231397543e+30;

nucleoid();

cursor1z = spawnerz + 1 \* 1;

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

vital\_components();

cursor1z = spawnerz + 1 \* 3.093963676e+29;

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

vital\_components2();

cursor1z = spawnerz + 1 \* 6.187927353e+29;

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

vital\_components3();

cursor1z = spawnerz + 1 \* 1;

cursor1x = spawnerx - 2 \* 6.187927353e+26;

cursor1y = spawnery - (1.23758547e+30 - 6.806720089e+28);

repeat\_endcell1 = Math.round(1.23758547e+30 / 6.497323721e+28 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(1.175706197e+30 / 3.712756412e+28 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

mitochondrion();

cursor1x = cursor1x - 3.712756412e+28;

}

cursor1x = cursor1x + 3.712756412e+28 \* Math.round(1.175706197e+30 / 3.712756412e+28 - 2);

cursor1z = cursor1z + 6.497323721e+28;

}

}

if (unused\_variable == 5) {

cursor2z = spawnerz + 1.23758547e+29;

cursor2y = spawnery - 1.732619659e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(17 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(16 \* 1 - 0);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

if (unused\_variable == 5) {

cursor2x = cursor2x - 1.23758547e+28;

lipid2 = 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2x = cursor2x + 1.23758547e+28;

lipid1 = lipid2;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

lipid2 = 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid\_wall\_x();

cursor2x = cursor2x - 1.23758547e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.23758547e+28;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 17;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.23758547e+29;

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(17 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 17;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.23758547e+29;

cursor2y = spawnery - (1.732619659e+29 + (6.187927353e+28 \* 15 + 1.23758547e+28));

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(17 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 17;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.23758547e+29;

cursor2y = spawnery - 1.732619659e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - 6.187927353e+26;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 6.187927353e+26;

lipid1 = 1.236347885e+30;

lipid\_wall\_x();

cursor2z = spawnerz + 6.187927353e+26;

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 1.231397543e+29;

lipid1 = 1.11320813e+30;

lipid\_wall\_x();

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.732619659e+29 + (6.187927353e+28 \* 15 + 1.23758547e+28));

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 1.23758547e+30 - (4.085906752e+25 + 2);

lipid1 = (1.236966678e+30 - (1.732619659e+29 + (6.187927353e+28 \* 15 + 1.23758547e+28))) - (4.085906752e+25 + 2);

lipid\_wall\_x();

cursor2z = spawnerz + (1.23758547e+29 + 6.187927353e+28 \* 16);

cursor2y = spawnery - 6.187927353e+26;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = (1.23758547e+30 - (1.23758547e+29 + 6.187927353e+28 \* 16)) - (4.085906752e+25 + 2);

lipid1 = 1.236347885e+30;

lipid\_wall\_x();

cursor2z = spawnerz + 1.23758547e+29;

cursor2y = spawnery - 6.187927353e+26;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = (1.23758547e+30 - 1.23758547e+29) - (4.085906752e+25 + 2);

lipid1 = 1.231397543e+29;

lipid\_wall\_x();

cursor2z = spawnerz + (1.23758547e+29 + 1.23758547e+28);

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - 1.175706197e+30;

repeat\_endcell1 = Math.round(16 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

lipid2 = 1.236966678e+30 - 1.23758547e+29;

lipid1 = 1.236966678e+29 - 1.23758547e+28;

lipid\_wall\_x();

}

cursor2z = cursor2z + 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.23758547e+29 + 0);

cursor2y = spawnery - 1.856378206e+29;

cursor2x = spawnerx - 1.175706197e+30;

repeat\_endcell1 = Math.round(15 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

lipid2 = (1.23758547e+30 - 1.23758547e+29) - (4.085906752e+25 + 2);

lipid1 = 4.950341882e+28;

lipid\_wall\_x();

}

cursor2y = cursor2y - 6.187927353e+28;

}

neuromuscular\_junction();

}

if (unused\_variable == 5) {

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - 1.732619659e+29;

cursor2x = spawnerx - 1.856378206e+29;

repeat\_endcell1 = Math.round(15 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(16 \* 1 - 0);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_z();

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid\_wall\_z();

lipid2 = 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid\_wall\_z();

lipid2 = 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2z = cursor2z - 3.588997865e+30;

lipid2 = 2.351412394e+30;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 4.950341882e+28;

lipid\_wall\_y();

cursor2z = cursor2z - 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid2 = 4.826583335e+28;

lipid\_wall\_x();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = Math.pow(((2.5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+28 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(4.950341882e+28 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(4.826583335e+28 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

salt\_calcium();

list\_molecules.push(['salt calcium', cursor1z, cursor1y, cursor1x, cursor1z + 1.92206873e+25, cursor1y - 1.92206873e+25, cursor1x - 1.92206873e+25]);

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2x = cursor2x - 1.23758547e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2x = cursor2x + 4.950341882e+28;

lipid1 = 4.950341882e+28;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2z = cursor2z + 3.588997865e+28;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2z = cursor2z - 3.588997865e+28;

cursor2x = cursor2x - 4.950341882e+28;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - 1.856378206e+29;

repeat\_endcell1 = Math.round(15 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_y();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - (1.23758547e+28 + (1.23758547e+29 + 15 \* 6.187927353e+28));

cursor2x = spawnerx - 1.856378206e+29;

repeat\_endcell1 = Math.round(15 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_y();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - (1.856378206e+29 - 4.950341882e+28);

repeat\_endcell1 = Math.round(16 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_x();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - 1.23758547e+29;

cursor2x = spawnerx - (1.856378206e+29 + (1.23758547e+28 + 14 \* 6.187927353e+28));

repeat\_endcell1 = Math.round(16 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_x();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.732619659e+29 + 4.950341882e+28);

cursor2x = spawnerx - (1.23758547e+29 - 4.950341882e+28);

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 4.331549147e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 4.950341882e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 200);

if (randomizer <= 50) {

peptidesequence = '1 mttqaptftq plqsvvvleg statfeahis gfpvpevswf rdgqvistst lpgvqisfsd 61 grakltipav tkansgrysl katngsgqat staellvkae tappnfvqrl qsmtvrqgsq 121 vrlqvrvtgi ptpvvkfyrd gaeiqssldf qisqegdlys lliaeayped sgtysvnatn 181 svgratstae llvqgeeevp akktktivst aqisesrqtr iekkieahfd arsiatvemv 241 idgaagqqlp hktpprippk pksrsptpps iaakaqlarq qspspirhsp spvrhvrapt 301 pspvrsvspa aristspirs vrspllmrkt qastvatgpe vpppwkqegy vassseaemr 361 ettlttstqi rteerwegry gvqeqvtisg aagaaasvsa sasyaaeava tgakevkqda 421 dksaavatvv aavdmarvre pvisaveqta qrttttavhi qpaqeqvrke aektavtkvv 481 vaadkakeqe lksrtkevit tkqeqmhvth eqirketekt fvpkvvisaa kakeqetris 541 eeitkkqkqv tqeairqete itaasmvvva takstkletv pgaqeetttq qdqmhlsyek 601 imketrktvv pkvivatpkv keqdlvsrgr egittkreqv qitqekmrke aektalstia 661 vatakakeqe tilrtretma trqeqiqvth gkvdvgkkae avatvvaavd qarvreprep 721 ghleesyaqq ttleygyker isaakvaepp qrpasephvv pkavkprviq apsethiktt 781 dqkgmhissq ikkttdltte rlvhvdkrpr tasphftvsk isvpktehgy easiagsaia 841 tlqkelsats saqkitksvk aptvkpsetr vraeptplpq fpfadtpdty kseagvevkk 901 evgvsitgtt vreerfevlh greakvteta rvpapveipv tpptlvsglk nvtviegesv 961 tlechisgyp sptvtwyred yqiessidfq itfqsgiarl mireafaeds grftcsavne 1021 agtvstscyl avqvseefek ettavtekft teekrfvesr dvvmtdtslt eeqagpgepa 1081 apyfitkpvv qklveggsvv fgcqvggnpk phvywkksgv plttgyrykv synkqtgeck 1141 lvismtfadd ageytivvrn khgetsasas lleeadyell mksqqemlyq tqvtafvqep 1201 kvgetapgfv yseyekeyek eqalirkkma kdtvvvrtyv edqefhissf eerlikeiey 1261 riikttleel leedgeekma vdiseseave sgfdsrikny rilegmgvtf hckmsgyplp 1321 kiawykdgkr ikhgeryqmd flqdgraslr ipvvlpedeg iytafasnik gnaicsgkly 1381 vepaaplgap tyiptlepvs rirslsprsv srspirmspa rmsparmspa rmsparmspg 1441 rrleetdesq lerlykpvfv lkpvsfkcle gqtarfdlkv vgrpmpetfw fhdgqqivnd 1501 ythkvviked gtqsliivpa tpsdsgewtv vaqnragrss isviltveav ehqvkpmfve 1561 klknvnikeg srlemkvrat gnpnpdivwl knsdiivphk ypkiriegtk geaalkidst 1621 vsqdsawyta tainkagrdt trckvnveve faepeperkl iiprgtyrak eiaapelepl 1681 hlrygqeqwe egdlydkekq qkpffkkklt slrlkrfgpa hfecrltpig dptmvvewlh 1741 dgkpleaanr lrminefgyc sldygvaysr dsgiitcrat nkygtdhtsa tlivkdeksl 1801 veesqlpegr kglqrieele rmahegaltg vttdqkekqk pdivlypepv rvlegetarf 1861 rcrvtgypqp kvnwylngql irkskrfrvr ydgihyldiv dcksydtgev kvtaenpegv 1921 iehkvkleiq qredfrsvlr rapeprpefh vhepgklqfe vqkvdrpvdt tetkevvklk 1981 raerithekv peeseelrsk fkrrteegyy eaitavelks rkkdesyeel lrktkdellh 2041 wtkelteeek kalaeegkit iptfkpdkie lspsmeapki feriqsqtvg qgsdahfrvr 2101 vvgkpdpece wykngvkier sdriywywpe dnvcelvird vtaedsasim vkainiaget 2161 sshafllvqa kqlitftqel qdvvakekdt matfecetse pfvkvkwykd gmevhegdky 2221 rmhsdrkvhf lsiltidtsd aedyscvlve denvkttakl ivegavvefv kelqdievpe 2281 sysgeleciv speniegkwy hndvelksng kytitsrrgr qnltvkdvtk edqgeysfvi 2341 dgkkttcklk mkprpiailq glsdqkvceg divqlevkvs lesvegvwmk dgqevqpsdr 2401 vhividkqsh mlliedmtke dagnysftip alglstsgrv svysvdvitp lkdvnviegt 2461 kavleckvsv pdvtsvkwyl ndeqikpddr vqaivkgtkq rlvinrthas degpyklivg 2521 rvetncnlsv ekikiirglr dltctetqnv vfevelshsg idvlwnfkdk eikpsskyki 2581 eahgkiyklt vlnmmkddeg kytfyagenm tsgkltvagg aiskpltdqt vaesqeavfe 2641 cevanpdskg ewlrdgkhlp ltnnirsesd ghkrrliiaa tklddigeyt ykvatsktsa 2701 klkveavkik ktlknltvte tqdavftvel thpnvkgvqw ikngvvlesn ekyaisvkgt 2761 iyslriknca ivdesvygfr lgrlgasarl hvetvkiikk pkdvtalena tvafevsvsh 2821 dtvpvkwfhk sveikpsdkh rlvserkvhk lmlqnispsd ageytavvgq leckaklfve 2881 tlhitktmkn ievpetktas fecevshfnv psmwlkngve iemsekfkiv vqgklhqlii 2941 mntstedsae ytfvcgndqv satltvtpim itsmlkdina eekdtitfev tvnyegisyk 3001 wlkngveiks tdkcqmrtkk lthslnirnv hfgdaadytf vagkatstat lyvearhief 3061 rkhikdikvl ekkramfece vsepditvqw mkddqelqit drikiqkeky vhrllipstr 3121 msdagkytvv aggnvstakl fvegrdvrir sikkevqvie kqravvefev neddvdahwy 3181 kdgieinfqv qerhkyvver rihrmfiset rqsdageytf vagrnrssvt lyvnapeppq 3241 vlqelqpvtv qsgkparfca visgrpqpki swykeeqlls tgfkckflhd gqeytlllie 3301 afpedaavyt ceakndygva ttsaslsvev pevvspdqem pvyppaiitp lqdtvtsegq 3361 parfqcrvsg tdlkvswysk dkkikpsrff rmtqfedtyq leiaeayped egtytfvasn 3421 avgqvsstan lsleapesil herieqeiem emkefsssfl saeeeglhsa elqlskinet 3481 lellsespvy ptkfdsekeg tgpifikevs nadismgdva tlsvtvigip kpkiqwffng 3541 vlltpsadyk fvfdgddhsl iilftklede geytcmasnd ygkticsayl kinskgeghk 3601 dtetesavak sleklggpcp phflkelkpi rcaqglpaif eytvvgepap tvtwfkenkq 3661 lctsvyytii hnpngsgtfi vndpqredsg lyickaenml gestcaaell vlledtdmtd 3721 tpckakstpe apedfpqtpl kgpavealds eqeiatfvkd tilkaalite enqqlsyehi 3781 akanelssql plgaqelqsi leqdkltpes treflcings ihfqplkeps pnlqlqivqs 3841 qktfskegil mpeepetqav lsdtekifps amsieqinsl tveplktlla epegnypqss 3901 ieppmhsylt svaeevlspk ektvsdtnre qrvtlqkqea qsalilsqsl aeghveslqs 3961 pdvmisqvny eplvpsehsc teggkilies anplenagqd savrieegks lrfplaleek 4021 qvllkeehsd nvvmppdqii eskrepvaik kvqevqgrdl lskesllsgi peeqrlnlki 4081 qicralqaav aseqpglfse wlrniekvev eavnitqepr himcmylvts aksvteevti 4141 iiedvdpqma nlkmelrdal caiiyeeidi ltaegpriqq gaktslqeem dsfsgsqkve 4201 pitepevesk ylisteevsy fnvqsrvkyl datpvtkgva savvsdekqd eslkpseeke 4261 esssesgtee vatvkiqeae gglikedgpm ihtplvdtvs eegdivhltt sitnakevnw 4321 yfenklvpsd ekfkclqdqn tytlvidkvn tedhqgeyvc ealndsgkta tsakltvvkr 4381 aapvikrkie plevalghla kftceiqsap nvrfqwfkag reiyesdkcs irsskyissl 4441 eilrtqvvdc geytckasne ygsvsctatl tvteaypptf lsrpkslttf vgkaakfict 4501 vtgtpvieti wqkdgaalsp spnwrisdae nkhilelsnl tiqdrgvysc kasnkfgadi 4561 cqaeliiidk phfikelepv qsainkkvhl ecqvdedrkv tvtwskdgqk lppgkdykic 4621 fedkiatlei plaklkdsgt yvctasneag ssscsatvtv reppsfvkkv dpsylmlpge 4681 sarlhcklkg spviqvtwfk nnkelsesnt vrmyfvnsea ilditdvkve dsgsysceav 4741 ndvgsdscst eivikeppsf iktlepadiv rgtnallqce vsgtgpfeis wfkdkkqirs 4801 skkyrlfsqk slvcleifsf nsadvgeyec vvanevgkcg cmathllkep ptfvkkvddl 4861 ialggqtvtl qaavrgsepi svtwmkgqev iredgkikms fsngvavlii pdvqisfggk 4921 ytclaeneag sqtsvgeliv kepakiiera eliqvtagdp atleytvagt pelkpkwykd 4981 grplvaskky risfknnvaq lkfysaelhd sgqytfeisn evgssscett ftvldrdiap 5041 fftkplrnvd svvngtcrld ckiagslpmr vswfkdgkei aasdryriaf vegtasleii 5101 rvdmndagnf tcratnsvgs kdssgalivq eppsfvtkpg skdvlpgsav clkstfqgst 5161 pltirwfkgn kelvsggscy itkealessl elylvktsds gtytckvsnv aggvecsanl 5221 fvkepatfve klepsqllkk gdatqlackv tgtppikitw fandreikes skhrmsfves 5281 tavlrltdvg iedsgeymce aqneagsdhc ssivivkesp yftkefkpie vlkeydvmll 5341 aevagtppfe itwfkdntil rsgrkyktfi qdhlvslqil kfvaadagey qcrvtnevgs 5401 sicsarvtlr eppsfikkie stsslrggta afqatlkgsl pitvtwlkds deiteddnir 5461 mtfennvasl ylsgievkhd gkyvcqaknd agiqrcsall svkepatite eavsidvtqg 5521 dpatlqvkfs gtkeitakwf kdgqeltlgs kykisvtdtv silkiistek kdsgeytfev 5581 qndvgrssck arinvldlii ppsftkklkk mdsikgsfid lecivagshp isiqwfkddq 5641 eisasekykf sfhdntafle isqlegtdsg tytcsatnka ghnqcsghlt vkeppyfvek 5701 pqsqdvnpnt rvqlkalvgg tapmtikwfk dnkelhsgaa rsvwkddtst slelfaakat 5761 dsgtyicqls ndvgtatska tlfvkeppqf ikkpspvlvl rngqsttfec qitgtpkirv 5821 swyldgneit aiqkhgisfi dglatfqisg arvensgtyv cearndagta scsielkvke 5881 pptfirelkp vevvkysdve lecevtgtpp fevtwlknnr eirsskkytl tdrvsvfnlh 5941 itkcdpsdtg eyqcivsneg gscscstrva lkeppsfikk ientttvlks satfqstvag 6001 sppisitwlk ddqildeddn vyisfvdsva tlqirsvdng hsgrytcqak nesgvercya 6061 fllvqepaqi vekaksvdvt ekdpmtlecv vagtpelkvk wlkdgkqivp sryfsmsfen 6121 nvasfriqsv mkqdsgqytf kvendfgsss cdaylrvldq nippsftkkl tkmdkvlgss 6181 ihmeckvsgs lpisaqwfkd gkeistsaky rlvchersvs levnnleled tanytckvsn 6241 vagddacsgi ltvkeppsfl vkpgrqqaip dstvefkail kgtppfkikw fkddvelvsg 6301 pkcfiglegs tsflnlysvd asktgqytch vtndvgsdsc ttmllvtepp kfvkkleask 6361 ivkagdssrl eckiagspei rvvwfrnehe lpasdkyrmt fidsvaviqm nnlstedsgd 6421 ficeaqnpag stscstkviv keppvfssfp pivetlknae vslecelsgt ppfevvwykd 6481 krqlrsskky kiasknfhts ihilnvdtsd igeyhckaqn evgsdtcvct vklkepprfv 6541 sklnsltvva gepaelqasi egaqpifvqw lkekeevire seniritfve nvatlqfaka 6601 epanagkyic qikndggmre nmatlmvlep avivekagpm tvtvgetctl eckvagtpel 6661 svewykdgkl ltssqkhkfs fynkisslri lsverqdagt ytfqvqnnvg kssctavvdv 6721 sdravppsft rrlkntggvl gascileckv agsspisvaw fhektkivsg akyqttfsdn 6781 vctlqlnsld ssdmgnytcv aanvagsdec ravltvqepp sfvkepeple vlpgknvtft 6841 svirgtppfk vnwfrgarel vkgdrcniyf edtvaelelf nidisqsgey tcvvsnnagq 6901 ascttrlfvk epaaflkrls dhsvepgksi ilestytgtl pisvtwkkdg fnittsekcn 6961 ivttektcil eilnstkrda gqysceiene agrdvcgalv stleppyfvt elepleaavg 7021 dsvslqcqva gtpeitvswy kgdtklrptp eyrtyftnnv atlvfnkvni ndsgeytcka 7081 ensigtassk tvfriqerql ppsfarqlkd ieqtvglpvt ltcrlngsap iqvcwyrdgv 7141 llrddenlqt sfvdnvatlk ilqtdlshsg qyscsasnpl gtasssarlt arepkkspff 7201 dikpvsidvi agesadfech vtgaqpmrit wskdnkeirp ggnytitcvg ntphlrilkv 7261 gkgdsgqytc qatndvgkdm csaqlsvkep pkfvkkleas kvakqgesiq leckisgspe 7321 ikvswfrnds elheswkynm sfinsvallt ineasaedsg dyiceahngv gdascstalt 7381 vkappvftqk pspvgalkgs dvilqceisg tppfevvwvk drkqvrnskk fkitskhfdt 7441 slhilnleas dvgeyhckat nevgsdtcsc svkfkepprf vkklsdtstl igdavelrai 7501 vegfqpisvv wlkdrgevir esentrisfi dniatlqlgs peasnsgkyi cqikndagmr 7561 ecsavltvle pariiekpep mtvttgnpfa lecvvtgtpe lsakwfkdgr elsadskhhi 7621 tfinkvaslk ipcaemsdkg lysfevknsv gksnctvsvh vsdrivppsf irklkdvnai 7681 lgasvvlecr vsgsapisvg wfqdgneivs gpkcqssfse nvctlnlsll epsdtgiytc 7741 vaanvagsde csavltvqep psfeqtpdsv evlpgmsltf tsvirgtppf kvkwfkgsre 7801 lvpgescnis ledfvtelel fevqplesgd ysclvtndag sasctthlfv kepatfvkrl 7861 adfsvetgsp ivleatytgt ppisvswikd eylisqserc sitmteksti leilestied 7921 yaqysclien eagqdiceal vsvleppyfi eplehveavi gepatlqckv dgtpeirisw 7981 ykehtklrsa paykmqfknn vaslvinkvd hsdvgeysck adnsvgavas savlvikark 8041 lppffarklk dvhetlgfpv afecringse plqvswykdg vllkddanlq tsfvhnvatl 8101 qilqtdqshi gqyncsasnp lgtasssakl ilsehevppf fdlkpvsvdl algesgtfkc 8161 hvtgtapiki twakdnreir pggnykmtlv entatltvlk vgkgdagqyt cyasniagkd 8221 scsaqlgvqe pprfikklep srivkqdeft ryeckiggsp eikvlwykde teiqesskfr 8281 msfvdsvavl emhnlsveds gdytceahna agsassstsl kvkeppifrk kphpietlkg 8341 advhlecelq gtppfhvswy kdkrelrsgk kykimsenfl tsihilnvda adigeyqcka 8401 tndvgsdtcv gsialkappr fvkklsdist vvgkevqlqt tiegaepisv vwfkdkgeiv 8461 resdniwisy seniatlqfs rvepanagky tcqikndagm qecfatlsvl epativekpe 8521 sikvttgdtc tlectvagtp elstkwfkdg keltsdnkyk isffnkvsgl kiinvapsds 8581 gvysfevqnp vgkdsctasl qvsdrtvpps ftrklketng lsgssvvmec kvygsppisv 8641 swfhegneis sgrkyqttlt dntcaltvnm leesdsgdyt ciatnmagsd ecsapltvre 8701 ppsfvqkpdp mdvltgtnvt ftsivkgtpp fsvswfkgss elvpgdrcnv sledsvaele 8761 lfdvdtsqsg eytcivsnea gkasctthly ikapakfvkr lndysiekgk plilegtftg 8821 tppisvtwkk nginvtpsqr cnittteksa ileipsstve dagqyncyie nasgkdscsa 8881 qilileppyf vkqlepvkvs vgdsaslqcq lagtpeigvs wykgdtklrp tttykmhfrn 8941 nvatlvfnqv dindsgeyic kaensvgevs astfltvqeq klppsfsrql rdvqetvglp 9001 vvfdcaisgs episvswykd gkplkdspnv qtsfldntat lnifktdrsl agqysctatn 9061 pigsasssar liltegknpp ffdirlapvd avvgesadfe chvtgtqpik vswakdsrei 9121 rsggkyqisy lensahltvl kvdkgdsgqy tcyavnevgk dsctaqlnik erlippsftk 9181 rlsetveete gnsfklegrv agsqpitvaw yknnieiqpt snceitfknn tlvlqvrkag 9241 mndaglytck vsndagsalc tssivikepk kppvfdqhlt pvtvsegeyv qlschvqgse 9301 piriqwlkag reikpsdrcs fsfasgtavl elrdvakads gdyvckasnv agsdttkskv 9361 tikdkpavap atkkaavdgr lffvsepqsi rvvekttatf iakvggdpip nvkwtkgkwr 9421 qlnqggrvfi hqkgdeakle irdttktdsg lyrcvafneh geiesnvnlq vderkkqeki 9481 egdlramlkk tpilkkgage eeeidimell knvdpkeyek yarmygitdf rgllqafell 9541 kqsqeeethr leieeierse rdekefeelv sfiqqrlsqt epvtlikdie nqtvlkdnda 9601 vfeidikiny peiklswykg teklepsdkf eisidgdrht lrvkncqlkd qgnyrlvcgp 9661 hiasakltvi epawerhlqd vtlkegqtct mtcqfsvpnv ksewfrngri lkpqgrhkte 9721 vehkvhklti advraedqgq ytckyedlet saelrieaep iqftkriqni vvsehqsatf 9781 ecevsfddai vtwykgptel tesqkynfrn dgrchymtih nvtpddegvy sviarleprg 9841 earstaelyl ttkeiklelk ppdipdsrvp iptmpiravp peeippvvap piplllptpe 9901 ekkpppkrie vtkkavkkda kkvvakpkem tpreeivkkp pppttlipak apeiidvssk 9961 aeevkimtit rkkevqkeke avyekkqavh kekrvfiesf eepydeleve pytepfeqpy 10021 yeepdedyee ikveakkevh eeweedfeeg qeyyereegy degeeeweea yqereviqvq 10081 kevyeesher kvpakvpekk appppkvikk pviekiekts rrmeeekvqv tkvpevskki 10141 vpqkpsrtpv qeevievkvp avhtkkmvis eekmffasht eeevsvtvpe vqkeivteek 10201 ihvaiskrve pppkvpelpe kpapeevapv pipkkveppa pkvpevpkkp vpeekkpvpv 10261 pkkepaappk vpevpkkpvp eekipvpvak kkeappakvp evqkgvvtee kitivtqree 10321 spppavpeip kkkvpeerkp vprkeeevpp ppkvpalpkk pvpeekvavp vpvakkappp 10381 raevskktvv eekrfvaeek lsfavpqrve vtrhevsaee ewsyseeeeg vsisvyreee 10441 reeeeeaevt eyevmeepee yvveeklhii skrveaepae vterqekkiv lkpkipakie 10501 epppakvpea pkkivpekkv papvpkkekv pppkvpeepk kpvpekkvpp kvikmeeplp 10561 akvterhmqi tqeekvlvav tkkeappkar vpeepkravp eekvlklkpk reeeppakvt 10621 efrkrvvkee kvsieapkre pqpikevtim eekeraytle eeavsvqree eyeeyeeydy 10681 kefeeyepte eydqyeeyee reyeryeehe eyitepekpi pvkpvpeepv ptkpkappak 10741 vlkkavpeek vpvpipkklk ppppkvpeep kkvfeekiri sitkrekeqv tepaakvpmk 10801 pkrvvaeekv pvprkevapp vrvpevpkel epeevafeee vvthveeylv eeeeeyihee 10861 eefiteeevv pvipvkvpev prkpvpeekk pvpvpkkkea ppakvpevpk kpeekvpvli 10921 pkkekpppak vpevpkkpvp eekvpvpvpk kveappakvp evpkkpvpek kvpvpapkkv 10981 eappakvpev pkklipeekk ptpvpkkvea pppkvpkkre pvpvpvalpq eeevlfeeei 11041 vpeeevlpee eevlpeeeev lpeeeevlpe eeeippeeee vppeeeyvpe eeefvpeeev 11101 lpevkpkvpv papvpeikkk vtekkvvipk keeappakvp evpkkveekr iilpkeeevl 11161 pvevteepee episeeeipe eppsieevee vapprvpevi kkavpeaptp vpkkveappa 11221 kvskkipeek vpvpvqkkea ppakvpevpk kvpekkvlvp kkeavppakg rtvleekvsv 11281 afrqevvvke rlelevveae veeipeeeef heveeyfeeg efheveefik leqhrveeeh 11341 rvekvhrvie vfeaeevevf ekpkappkgp eisekiippk kpptkvvprk eppakvpevp 11401 kkivveekvr vpeeprvppt kvpevlppke vvpekkvpvp pakkpeappp kvpeapkevv 11461 pekkvpvppp kkpevpptkv pevpkaavpe kkvpeaippk pespppevpe apkevvpekk 11521 vpaappkkpe vtpvkvpeap kevvpekkvp vpppkkpevp ptkvpevpkv avpekkvpea 11581 ippkpesppp evfeepeeva leeppaevve epepaappqv tvppkkpvpe kkapavvakk 11641 pelppvkvpe vpkevvpekk vplvvpkkpe appakvpevp kevvpekkva vpkkpevppa 11701 kvpevpkkpv leekpavpvp eraespppev yeepeeiape eeiapeeekp vpvaeeeepe 11761 vpppavpeep kkiipekkvp vikkpeappp kepepekvie kpklkprppp pppappkedv 11821 kekifqlkai pkkkvpekpq vpekveltpl kvpggekkvr kllperkpep keevvlksvl 11881 rkrpeeeepk vepkklekvk kpavpepppp kpveevevpt vtkrerkipe ptkvpeikpa 11941 iplpapepkp kpeaevktik pppvepeptp iaapvtvpvv gkkaeakapk eeaakpkgpi 12001 kgvpkktpsp ieaerrklrp gsggekppde apftyqlkav plkfvkeikd iiltesefvg 12061 ssaifeclvs pstaittwmk dgsnirespk hrfiadgkdr klhiidvqls dageytcvlr 12121 lgnkektsta klvveelpvr fvktleeevt vvkgqplyls celnkerdvv wrkdgkivve 12181 kpgrivpgvi glmraltind addtdagtyt vtvenannle csscvkvvev irdwlvkpir 12241 dqhvkpkgta ifacdiakdt pnikwfkgyd eipaepndkt eilrdgnhly lkiknamped 12301 iaeyaveieg krypakltlg erevellkpi edvtiyekes asfdaeisea dipgqwklkg 12361 ellrpsptce ikaeggkrfl tlhkvkldqa gevlyqalna ittailtvke ieldfavplk 12421 dvtvperrqa rfecvltrea nviwskgpdi ikssdkfdii adgkkhilvi ndsqfddegv 12481 ytaevegkkt sarlfvtgir lkfmspledq tvkegetatf vcelshekmh vvwfkndakl 12541 htsrtvliss egkthklemk evtlddisqi kaqvkelsst aqlkvleadp yftvklhdkt 12601 avekdeitlk cevskdvpvk wfkdgeeivp spkysikadg lrrilkikka dlkdkgeyvc 12661 dcgtdktkan vtvearlikv ekplygvevf vgetahfeie lsepdvhgqw klkgqpltas 12721 pdceiiedgk khililhncq lgmtgevsfq aanaksaanl kvkelplifi tplsdvkvfe 12781 kdeakfecev srepktfrwl kgtqeitgdd rfelikdgtk hsmviksaaf edeakymfea 12841 edkhtsgkli iegirlkflt plkdvtakek esavftvels hdnirvkwfk ndqrlhttrs 12901 vsmqdegkth sitfkdlsid dtsqirveam gmsseakltv legdpyftgk lqdytgvekd 12961 evilqceisk adapvkwfkd gkeikpskna vikadgkkrm lilkkalksd igqytcdcgt 13021 dktsgkldie dreiklvrpl hsvevmetet arfeteised dihanwklkg eallqtpdce 13081 ikeegkihsl vlhncrldqt ggvdfqaanv kssahlrvkp rvigllrplk dvtvtageta 13141 tfdcelsyed ipvewylkgk klepsdkvvp rsegkvhtlt lrdvkledag evqltakdfk 13201 thanlfvkep pveftkpled qtveegatav lecevsrena kvkwfkngte ilkskkyeiv 13261 adgrvrklvi hdctpedikt ytcdakdfkt scnlnvvpph veflrpltdl qvrekemarf 13321 ecelsrenak vkwfkdgaei kkgkkydiis kgavrilvin kcllddeaey scevrtarts 13381 gmltvleeea vftknlanie vsetdtiklv cevskpgaev iwykgdeeii etgryeilte 13441 grkrilviqn ahledagnyn crlpssrtdg kvkvhelaae fiskpqnlei legekaefvc 13501 siskesfpvq wkrddktles gdkydviadg kkrvlvvkda tlqdmgtyvv mvgaaraaah 13561 ltvieklriv vplkdtrvke qqevvfncev ntegakakwf rneeaifdss kyiilqkdlv 13621 ytlrirdahl ddqanynvsl tnhrgenvks aanliveeed lriveplkdi etmekksvtf 13681 wckvnrlnvt lkwtkngeev pfdnrvsyrv dkykhmltik dcgfpdegey ivtagqdksv 13741 aelliieapt efvehledqt vtefddavfs cqlsrekanv kwyrngreik egkkykfekd 13801 gsihrliikd crlddeceya cgvedrksra rlfveeipve iirppqdile apgadvvfla 13861 elnkdkvevq wlrnnmvvvq gdkhqmmseg kihrlqicdi kprdqgeyrf iakdkearak 13921 lelaaapkik tadqdlvvdv gkpltmvvpy daypkaeaew fkeneplstk tidttaeqts 13981 frileakkgd kgrykivlqn khgkaegfin lkvidvpgpv rnlevtetfd gevslaweep 14041 ltdggskiig yvverrdikr ktwvlatdra esceftvtgl qkggveylfr vsarnrvgtg 14101 epvetdnpve arskydvpgp plnvtitdvn rfgvsltwep peydggaeit nyvielrdkt 14161 sirwdtamtv raedlsatvt dvvegqeysf rvraqnrigv gkpsaatpfv kvadpierps 14221 ppvnltssdq tqssvqlkwe pplkdggspi lgyiiercee gkdnwircnm klvpeltykv 14281 tglekgnkyl yrvsaenkag vsdpseilgp ltaddafvep tmdlsafkdg levivpnpit 14341 ilvpstgypr ptatwcfgdk vletgdrvkm ktlsayaelv ispsersdkg iytlklenrv 14401 ktisgeidvn viarpsapke lkfgditkds vhltweppdd dggspltgyv vekrevsrkt 14461 wtkvmdfvtd leftvpdlvq gkeylfkvca rnkcgpgepa yvdepvnmst patvpdppen 14521 vkwrdrtans ifltwdppkn dggsrikgyi vercprgsdk wvacgepvae tkmevtglee 14581 gkwyayrvka lnrqgaskps rpteeiqavd tqeapeifld vkllagltvk agtkielpat 14641 vtgkpepkit wtkadmilkq dkritienvp kkstvtivds krsdtgtyii eavnvcgrat 14701 avvevnvldk pgppaafdit dvtnescllt wnpprddggs kitnyvverr atdsevwhkl 14761 sstvkdtnfk atklipnkey ifrvaaenmy gvgepvqasp itakyqfdpp gpptrlepsd 14821 itkdavtltw cepdddggsp itgywverld pdtdkwvrcn kmpvkdttyr vkgltnkkky 14881 rfrvlaenla gpgkpskste pilikdpidp pwppgkptvk dvgktsvrln wtkpehdgga 14941 kiesyvieml ktgtdewvrv aegvpttqhl lpglmegqey sfrvravnka gesepsepsd 15001 pvlcreklyp pspprwlevi nitkntadlk wtvpekdggs pitnyivekr dvrrkgwqtv 15061 dttvkdtkct vtpltegsly vfrvaaenai gqsdyteied svlakdtftt pgppyalavv 15121 dvtkrhvdlk weppkndggr piqryviekk erlgtrwvka gktagpdcnf rvtdviegte 15181 vqfqvraene agvghpsept eilsiedpts ppsppldlhv tdagrkhiai awkppekngg 15241 spiigyhvem cpvgtekwmr vnsrpikdlk fkveegvvpd keyvlrvrav naigvsepse 15301 isenvvakdp dckptidlet hdiiviegek lsipvpfrav pvptvswhkd gkevkasdrl 15361 tmkndhisah levpksvrad agiytitlen klgsatasin vkviglpgpc kdikasditk 15421 ssckltwepp efdggtpilh yvlerreagr rtyipvmsge nklswtvkdl ipngeyffrv 15481 kavnkvggge yielknpvia qdpkqppdpp vdvevhnpta eamtitwkpp lydggskimg 15541 yiiekiakge erwkrcnehl vpiltytakg leegkeyqfr vraenaagis epsratpptk 15601 avdpidapkv ilrtslevkr gdeialdasi sgspyptitw ikdenvivpe eikkraaplv 15661 rrrkgevqee epfvlpltqr lsidnskkge sqlrvrdslr pdhglymikv endhgiakap 15721 ctvsvldtpg ppinfvfedi rktsvlckwe pplddggsei inytlekkdk tkpdsewivv 15781 tstlrhckys vtkliegkey lfrvraenrf gpgppcvskp lvakdpfgpp dapdkpived 15841 vtsnsmlvkw nepkdngspi lgywlekrev nsthwsrvnk sllnalkanv dgllegltyv 15901 frvcaenaag pgkfsppsdp ktahdpispp gppiprvtdt ssttielewe ppafngggei 15961 vgyfvdkqlv gtnewsrcte kmikvrqytv keiregadyk lrvsavnaag egppgetqpv 16021 tvaepqeppa veldvsvkgg iqimagktlr ipavvtgrpv ptkvwtkeeg eldkdrvvid 16081 nvgtkselii kdalrkdhgr yvitatnscg skfaaarvev fdvpgpvldl kpvvtnrkmc 16141 llnwsdpedd ggseitgfii erkdakmhtw rqpietersk cditgllegq eykfrviakn 16201 kfgcgppvei gpilavdplg pptsperlty tertkstitl dwkeprsngg spiqgyiiek 16261 rrhdkpdfer vnkrlcptts flvenldehq myefrvkavn eigesepslp lnvviqddev 16321 pptiklrlsv rgdtikvkag epvhipadvt glpmpkiews knetviekpt dalqitkeev 16381 srseaktels ipkavredkg tytvtasnrl gsvfrnvhve vydrpspprn lavtdikaes 16441 cyltwdapld nggseithyv idkrdasrkk aeweevtnta vekrygiwkl ipngqyefrv 16501 ravnkygisd ecksdkvviq dpyrlpgppg kpkvlartkg smlvswtppl dnggspitgy 16561 wlekreegsp ywsrvsrapi tkvglkgvef nvprllegvk yqframaina agigppseps 16621 dpevagdpif ppgppscpev kdktkssisl gwkppakdgg spikgyivem qeegttdwkr 16681 vnepdklitt cecvvpnlke lrkyrfrvka vneageseps dttgeipatd iqeepevfid 16741 igaqdclvck agsqiripav ikgrptpkss wefdgkakka mkdgvhdipe daqletaens 16801 sviiipeckr shtgkysita knkagqktan crvkvmdvpg ppkdlkvsdi trgscrlswk 16861 mpdddggdri kgyviekrti dgkawtkvnp dcgsttfvvp dllseqqyff rvraenrfgi 16921 gppvetiqrt tardpiyppd ppiklkigli tkntvhlswk ppkndggspv thyiveclaw 16981 dptgtkkeaw rqcnkrdvee lqftvedlve ggeyefrvka vnaagvskps atvgpvtvkd 17041 qtcppsidlk efmeveegtn vnivakikgv pfptltwfka ppkkpdnkep vlydthvnkl 17101 vvddtctlvi pqsrrsdtgl ytitavnnlg taskemrlnv lgrpgppvgp ikfesvsadq 17161 mtlswfppkd dggskitnyv iekreanrkt wvhvssepke ctytipklle gheyvfrima 17221 qnkygigepl dsepetarnl fsvpgapdkp tvssvtrnsm tvnweepeyd ggspvtgywl 17281 emkdttskrw krvnrdpika mtlgvsykvt gliegsdyqf rvyainaagv gpaslpsdpa 17341 tardpiappg ppfpkvtdwt kssadlewsp plkdggskvt gyiveykeeg keewekgkdk 17401 evrgtklvvt glkegafykf rvravniagi gepgevtdvi emkdrlvspd lqldasvrdr 17461 ivvhaggvir iiayvsgkpp ptvtwnmner tlpqeatiet taisssmvik ncqrshqgvy 17521 sllakneage rkktiivdvl dvpgpvgtpf lahnltnesc kltwfspedd ggspitnyvi 17581 ekresdrraw tpvtytvtrq natvqgliqg kayffriaae nsigmgpfve tsealvirep 17641 itvperpedl evkevtkntv tltwnppkyd ggseiinyvl esrligtekf hkvtndnlls 17701 rkytvkglke gdtyeyrvsa vnivgqgkps fctkpitckd elapptlhld frdkltirvg 17761 eafaltgrys gkpkpkvswf kdeadvledd rthikttpat lalekikakr sdsgkycvvv 17821 enstgsrkgf cqvnvvdrpg ppvgpvsfde vtkdymvisw kpplddggsk itnyiiekke 17881 vgkdvwmpvt sasakttckv skllegkdyi frihaenlyg isdplvsdsm kakdrfrvpd 17941 apdqpivtev tkdsalvtwn kphdggkpit nyilekretm skrwarvtkd pihpytkfrv 18001 pdllegcqye frvsaeneig igdpsppskp vfakdpiakp sppvnpeaid ttcnsvdltw 18061 qpprhdggsk ilgyiveyqk vgdeewrran htpescpetk ykvtglrdgq tykfrvlavn 18121 aagesdpahv pepvlvkdrl eppelildan mareqhikvg dtlrlsaiik gvpfpkvtwk 18181 kedrdaptka ridvtpvgsk leirnaahed ggiysltven pagsktvsvk vlvldkpgpp 18241 rdlevseirk dscyltwkep lddggsvitn yvverrdvas aqwsplsats kkkshfakhl 18301 negnqylfrv aaenqygrgp fvetpkpika ldplhppgpp kdlhhvdvdk tevslvwnkp 18361 drdggspitg ylveyqeegt qdwikfktvt nlecvvtglq qgktyrfrvk aenivglglp 18421 dttipiecqe klvppsveld vklieglvvk agttvrfpai irgvpvptak wttdgseikt 18481 dehytvetdn fssvltiknc lrrdtgeyqi tvsnaagskt vavhltvldv pgpptgpini 18541 ldvtpehmti swqppkddgg spvinyivek qdtrkdtwgv vssgssktkl kiphlqkgce 18601 yvfrvraenk igvgppldst ptvakhkfsp psppgkpvvt ditenaatvs wtlpksdggs 18661 pitgyymerr evtgkwvrvn ktpiadlkfr vtglyegnty efrvfaenla glskpspssd 18721 pikacrpikp pgppinpklk dksretadlv wtkplsdggs pilgyvvecq kpgtaqwnri 18781 nkdelirqca frvpgliegn eyrfrikaan ivgegeprel aesviakdil hppeveldvt 18841 crdvitvrvg qtirilarvk grpepditwt kegkvlvrek rvdliqdlpr velqikeavr 18901 adhgkyiisa knssghaqgs aivnvldrpg pcqnlkvtnv tkenctiswe npldnggsei 18961 tnfiveyrkp nqkgwsivas dvtkrlikan llanneyyfr vcaenkvgvg ptietktpil 19021 ainpidrpge penlhiadkg ktfvylkwrr pdydggspnl syhverrlkg sddwervhkg 19081 sikethymvd rcvenqiyef rvqtknegge sdwvkteevv vkedlqkpvl dlklsgvltv 19141 kagdtirlea gvrgkpfpev awtkdkdatd ltrsprvkid tradsskfsl tkakrsdggk 19201 yvvtatntag sfvayatvnv ldkpgpvrnl kivdvssdrc tvcwdppedd ggceiqnyil 19261 ekcetkrmvw stysatvltp gttvtrlieg neyifrvrae nkigtgppte skpviaktky 19321 dkpgrpdppe vtkvskeemt vvwnppeydg gksitgyfle kkekhstrwv pvnksaiper 19381 rmkvqnllpd heyqfrvkae neigigepsl psrpvvakdp ieppgpptnf rvvdttkhsi 19441 tlgwgkpvyd ggapiigyvv emrpkiadas pdegwkrcna aaqlvrkeft vtsldenqey 19501 efrvcaqnqv gigrpaelke aikpkeilep peidldasmr klvivragcp irlfaivrgr 19561 papkvtwrkv gidnvvrkgq vdlvdtmafl vipnstrdds gkysltlvnp agekavfvnv 19621 rvldtpgpvs dlkvsdvtkt schvswappe ndggsqvthy ivekreadrk twstvtpevk 19681 ktsfhvtnlv pgneyyfrvt avneygpgvp tdvpkpvlas dplsepdppr klevtemtkn 19741 satlawlppl rdggakidgy itsyreeeqp adrwteysvv kdlslvvtgl kegkkykfrv 19801 aarnavgvsl preaegvyea keqllppkil mpeqitikag kklrieahvy gkphptckwk 19861 kgedevvtss hlavhkadss siliikdvtr kdsgyyslta enssgtdtqk ikvvvmdapg 19921 ppqppfdisd idadacslsw hipledggsn itnyivekcd vsrgdwvtal asvtktscrv 19981 gklipgqeyi frvraenrfg isepltspkm vaqfpfgvps epknarvtkv nkdcifvawd 20041 rpdsdggspi igylierker nsllwvkand tlvrsteypc aglvegleys friyalnkag 20101 ssppskptey vtarmpvdpp gkpevidvtk stvsliwarp khdggskiig yfveacklpg 20161 dkwvrcntap hqipqeeyta tgleekaqyq fraiartavn isppsepsdp vtilaenvpp 20221 ridlsvamks lltvkagtnv cldatvfgkp mptvswkkdg tllkpaegik mamqrnlctl 20281 elfsvnrkds gdytitaens sgsksatikl kvldkpgppa svkinkmysd ramlsweppl 20341 edggseitny ivdkretsrp nwaqvsatvp itscsvekli egheyqfric aenkygvgdp 20401 vftepaiakn pydppgrcdp pvisnitkdh mtvswkppad dggspitgyl lekretqavn 20461 wtkvnrkpii ertlkatglq egteyefrvt ainkagpgkp sdaskaayar dpqyppgppa 20521 fpkvydttrs svslswgkpa ydggspiigy lvevkradsd nwvrcnlpqn lqktrfevtg 20581 lmedtqyqfr vyavnkigys dpsdvpdkhy pkdilippeg eldadlrktl ilragvtmrl 20641 yvpvkgrppp kitwskpnvn lrdrigldik stdfdtflrc envnkydagk yiltlenscg 20701 kkeytivvkv ldtpgppvnv tvkeiskdsa yvtweppiid ggspiinyvv qkrdaerksw 20761 stvttecskt sfrvanleeg ksyffrvfae neygigdpge trdavkasqt pgpvvdlkvr 20821 svsksscsig wkkphsdggs riigyvvdfl teenkwqrvm kslslqysak dltegkeytf 20881 rvsaenenge gtpseitvva rddvvapdld lkglpdlcyl akensnfrlk ipikgkpaps 20941 vswkkgedpl atdtrvsves savnttlivy dcqksdagky titlknvagt kegtisikvv 21001 gkpgiptgpi kfdevtaeam tlkwappkdd ggseitnyil ekrdsvnnkw vtcasavqkt 21061 tfrvtrlheg meytfrvsae nkygvgeglk sepivarhpf dvpdappppn ivdvrhdsvs 21121 ltwtdpkktg gspitgyhle fkernsllwk ranktpirmr dfkvtglteg leyefrvmai 21181 nlagvgkpsl psepvvaldp idppgkpevi nitrnsvtli wtepkydggh kltgyivekr 21241 dlpskswmka nhvnvpecaf tvtdlveggk yefriraknt agaisapses tetiickdey 21301 eaptivldpt ikdgltikag dtivlnaisi lgkplpkssw skagkdirps ditqitstpt 21361 ssmltikyat rkdageytit atnpfgtkve hvkvtvldvp gppgpveisn vsaekatltw 21421 tppledggsp iksyilekre tsrllwtvvs ediqscrhva tkliqgneyi frvsavnhyg 21481 kgepvqsepv kmvdrfgppg ppekpevsnv tkntatvswk rpvddggsei tgyhverrek 21541 kslrwvraik tpvsdlrckv tglqegstye frvsaenrag igppseasds vlmkdaaypp 21601 gppsnphvtd ttkksaslaw gkphydggle itgyvvehqk vgdeawikdt tgtalritqf 21661 vvpdlqtkek ynfrisaind agvgepavip dveiverema pdfeldaelr rtlvvragls 21721 irifvpikgr papevtwtkd ninlknrani entesftlli ipecnrydtg kfvmtienpa 21781 gkksgfvnvr vldtpgpvln lrptditkds vtlhwdlpli dggsritnyi vekreatrks 21841 ystattkchk ctykvtglse gceyffrvma eneygigept ettepvkase apsppdslni 21901 mditkstvsl awpkpkhdgg skitgyviea qrkgsdqwth ittvkglecv vrnltegeey 21961 tfqvmavnsa grsapresrp vivkeqtmlp eldlrgiyqk lviakagdni kveipvlgrp 22021 kptvtwkkgd qilkqtqrvn fettatstil ninecvrsds gpypltarni vgevgdviti 22081 qvhdipgppt gpikfdevss dfvtfswdpp endggvpisn yvvemrqtds ttwvelattv 22141 irttykatrl ttgleyqfrv kaqnrygvgp gitsacivan ypfkvpgppg tpqvtavtkd 22201 smtiswhepl sdggspilgy hverkerngi lwqtvskalv pgnifkssgl tdgiayefrv 22261 iaenmagksk pskpsepmla ldpidppgkp vplnitrhtv tlkwakpeyt ggfkitsyiv 22321 ekrdlpngrw lkanfsnile neftvsglte daayefrvia knaagaispp sepsdaitcr 22381 ddveapkikv dvkfkdtvil kageafrlea dvsgrppptm ewskdgkele gtakleikia 22441 dfstnlvnkd strrdsgayt ltatnpggfa khifnvkvld rpgppegpla vtevtsekcv 22501 lswfpplddg gakidhyivq kretsrlawt nvasevqvtk lkvtkllkgn eyifrvmavn 22561 kygvgeples epvlavnpyg ppdppknpev ttitkdsmvv cwghpdsdgg seiinyiver 22621 rdkagqrwik cnkktltdlr ykvsgltegh eyefrimaen aagisapspt spfykacdtv 22681 fkpgppgnpr vldtsrssis iawnkpiydg gseitgymve ialpeedewq ivtppaglka 22741 tsytitglte nqeykiriya mnseglgepa lvpgtpkaed rmlppeield adlrkvvtir 22801 acctlrlfvp ikgrpapevk wardhgesld kasiestssy tllivgnvnr fdsgkyiltv 22861 enssgsksaf vnvrvldtpg ppqdlkvkev tktsvtltwd pplldggski knyivekres 22921 trkaystvat nchktswkvd qlqegcsyyf rvlaeneygi glpaetaesv kaserplppg 22981 kitlmdvtrn svslswekpe hdggsrilgy ivemqtkgsd kwatcatvkv teatitgliq 23041 geeysfrvsa qnekgisdpr qlsvpviakd lvippafkll fntftvlage dlkvdvpfig 23101 rptpavtwhk dnvplkqttr vnaestenns lltikdacre dvghyvvklt nsageaietl 23161 nvivldkpgp ptgpvkmdev tadsitlswg ppkydggssi nnyivekrdt stttwqivsa 23221 tvarttikac rlktgceyqf riaaenrygk stylnseptv aqypfkvpgp pgtpvvtlss 23281 rdsmevqwne pisdggsrvi gyhlerkern silwvklnkt pipqtkfktt gleegveyef 23341 rvsaenivgi gkpskvsecy vardpcdppg rpeaiivtrn svtlqwkkpt ydggskitgy 23401 ivekkelpeg rwmkasftni idthfevtgl vedhryefrv iarnaagvfs epsestgait 23461 ardevdppri smdpkykdti vvhagesfkv dadiygkpip tiqwikgdqe lsntarleik 23521 stdfatslsv kdavrvdsgn yilkaknvag ersvtvnvkv ldrpgppegp vvisgvtaek 23581 ctlawkpplq dggsdiinyi verretsrlv wtvvdanvqt lsckvtklle gneytfrima 23641 vnkygvgepl esepvvaknp fvvpdapkap evttvtkdsm ivvwerpasd ggseilgyvl 23701 ekrdkegirw trchkrlige lrlrvtglie nhdyefrvsa enaaglseps ppsayqkacd 23761 piykpgppnn pkviditrss vflswskpiy dggceiqgyi vekcdvsvge wtmctpptgi 23821 nktnievekl lekheynfri cainkagvge hadvpgpiiv eekleapdid ldlelrkiin 23881 iraggslrlf vpikgrptpe vkwgkvdgei rdaaiidvts sftslvldnv nrydsgkytl 23941 tlenssgtks afvtvrvldt psppvnlkvt eitkdsvsit wepplldggs kiknyivekr 24001 eatrksyaav vtnchknswk idqlqegcsy yfrvtaeney giglpaqtad pikvaevpqp 24061 pgkitvddvt rnsvslswtk pehdggskii qyivemqakh sekwsecarv kslqavitnl 24121 tqgeeylfrv vavnekgrsd prslavpiva kdlviepdvk pafssysvqv gqdlkievpi 24181 sgrpkptitw tkdglplkqt trinvtdsld lttlsiketh kddggqygit vanvvgqkta 24241 sieivtldkp dppkgpvkfd dvsaesitls wnpplytggc qitnyivqkr dttttvwdvv 24301 satvarttlk vtklktgtey qfrifaenry gqsfalesdp ivaqypykep gppgtpfata 24361 iskdsmviqw hepvnnggsp vigyhlerke rnsilwtkvn ktiihdtqfk aqnleegiey 24421 efrvyaeniv gvgkasknse cyvardpcdp pgtpepimvk rneitlqwtk pvydggsmit 24481 gyivekrdlp dgrwmkasft nvietqftvs gltedqryef rviaknaaga iskpsdstgp 24541 itakdevelp rismdpkfrd tivvnagetf rleadvhgkp lptiewlrgd keieesarce 24601 ikntdfkall ivkdairidg gqyilrasnv agsksfpvnv kvldrpgppe gpvqvtgvts 24661 ekcsltwspp lqdggsdish yvvekretsr lawtvvasev vtnslkvtkl legneyvfri 24721 mavnkygvge plesapvlmk npfvlpgppk slevtniakd smtvcwnrpd sdggseiigy 24781 ivekrdrsgi rwikcnkrri tdlrlrvtgl tedheyefrv saenaagvge pspatvyyka 24841 cdpvfkpgpp tnahivdttk nsitlawgkp iydggseilg yvveickade eewqivtpqt 24901 glrvtrfeis kltehqeyki rvcalnkvgl geatsvpgtv kpedkleape ldldselrkg 24961 ivvraggsar ihipfkgrpt peitwsreeg eftdkvqiek gvnytqlsid ncdrndagky 25021 ilklenssgs ksafvtvkvl dtpgppqnla vkevrkdsaf lvweppiidg gakvknyvid 25081 krestrkaya nvsskcskts fkvenltega iyyfrvmaen efgvgvpvet vdavkaaepp 25141 sppgkvtltd vsqtsaslmw ekpehdggsr vlgyvvemqp kgtekwsiva eskvcnavvt 25201 glssgqeyqf rvkaynekgk sdprvlgvpv iakdltiqps lklpfntysi qagedlkiei 25261 pvigrprpni swvkdgeplk qttrvnveet atstvlhike gnkddfgkyt vtatnsagta 25321 tenlsvivle kpgppvgpvr fdevsadfvv isweppaytg gcqisnyive krdtttttwh 25381 mvsatvartt ikitklktgt eyqfrifaen rygksaplds kavivqypfk epgppgtpfv 25441 tsiskdqmlv qwhepvndgg tkiigyhleq keknsilwvk lnktpiqdtk fkttgldegl 25501 eyefkvsaen ivgigkpskv secfvardpc dppgrpeaiv itrnnvtlkw kkpaydggsk 25561 itgyivekkd lpdgrwmkas ftnvleteft vsglvedqry efrviarnaa gnfsepsdss 25621 gaitardeid apnasldpky kdvivvhage tfvleadirg kpipdvvwsk dgkeleetaa 25681 rmeikstiqk ttlvvkdcir tdggqyilkl snvggtksip itvkvldrpg ppegplkvtg 25741 vtaekcylaw npplqdggan ishyiiekre tsrlswtqvs tevqalnykv tkllpgneyi 25801 frvmavnkyg igeplesgpv tacnpykppg ppstpevsai tkdsmvvtwa rpvddggtei 25861 egyilekrdk egvrwtkcnk ktltdlrlrv tglteghsye frvaaenaag vgepsepsvf 25921 yracdalypp gppsnpkvtd tsrssvslaw skpiydggap vkgyvvevke aaadewttct 25981 pptglqgkqf tvtklkente ynfricains egvgepatlp gsvvaqerie ppeieldadl 26041 rkvvvlrasa tlrlfvtikg rpepevkwek aegiltdraq ievtssftml vidnvtrfds 26101 grynltlenn sgsktafvnv rvldspsapv nltirevkkd svtlsweppl idggakitny 26161 ivekrettrk ayatitnnct kttfrienlq egcsyyfrvl asneygiglp aettepvkvs 26221 epplppgrvt lvdvtrntat ikwekpesdg gskitgyvve mqtkgsekws tctqvktlea 26281 tisgltagee yvfrvaavne kgrsdprqlg vpviardiei kpsvelpfht fnvkareqlk 26341 idvpfkgrpq atvnwrkdgq tlkettrvnv sssktvtsls ikeaskedvg tyelcvsnsa 26401 gsitvpitii vldrpgppgp iridevscds itiswnppey dggcqisnyi vekkettstt 26461 whivsqavar tsikivrltt gseyqfrvca enrygkssys essavvaeyp fsppgppgtp 26521 kvvhatkstm lvtwqvpvnd ggsrvigyhl eykerssilw skankiliad tqmkvsglde 26581 glmyeyrvya eniagigkcs kscepvpard pcdppgqpev tnitrksvsl kwskphydgg 26641 akitgyiver relpdgrwlk cnytniqety fevteltedq ryefrvfarn aadsvsepse 26701 stgpiivkdd vepprvmmdv kfrdvivvka gevlkinadi agrplpvisw akdgieieer 26761 arteiistdn htlltvkdci rrdtgqyvlt lknvagtrsv avnckvldkp gppagplein 26821 gltaekcsls wgrpqedgga didyyivekr etshlawtic egelqmtsck vtkllkgney 26881 ifrvtgvnky gvgeplesva ikaldpftvp spptsleits vtkesmtlcw srpesdggse 26941 isgyiierre knslrwvrvn kkpvydlrvk stglregcey eyrvyaenaa glslpsetsp 27001 liraedpvfl psppskpkiv dsgkttitia wvkplfdgga pitgytveyk ksddtdwkts 27061 iqslrgteyt isglttgaey vfrvksvnkv gasdpsdssd pqiakereee plfdidsemr 27121 ktlivkagas ftmtvpfrgr pvpnvlwskp dtdlrtrayv dttdsrtslt ienanrndsg 27181 kytltiqnvl saasltlvvk vldtpgpptn itvqdvtkes avlswdvpen dggapvknyh 27241 iekreaskka wvsvtnncnr lsykvtnlqe gaiyyfrvsg enefgvgipa etkegvkite 27301 kpsppeklgv tsiskdsvsl twlkpehdgg srivhyvvea lekgqknwvk cavaksthhv 27361 vsglrensey ffrvfaenqa glsdprelll pvlikeqlep peidmknfps htvyvragsn 27421 lkvdipisgk plpkvtlsrd gvplkatmrf nteitaenlt inlkesvtad agryeitaan 27481 ssgttkafin ivvldrpgpp tgpvvisdit eesvtlkwep pkydggsqvt nyillkrets 27541 tavwtevsat vartmmkvmk lttgeeyqfr ikaenrfgis dhidsacvtv klpyttpgpp 27601 stpwvtnvtr esitvgwhep vsnggsavvg yhlemkdrns ilwqkanklv irtthfkvtt 27661 isagliyefr vyaenaagvg kpshpsepvl aidacepprn vritdiskns vslswqqpaf 27721 dggskitgyi verrdlpdgr wtkasftnvt etqfiisglt qnsqyefrvf arnavgsisn 27781 psevvgpitc idsyggpvid lpleytevvk yragtsvklr agisgkpapt iewykddkel 27841 qtnalvcven ttdlasilik dadrlnsgcy elklrnamgs asatirvqil dkpgppggpi 27901 efktvtaeki tllwrppadd ggakithyiv ekretsrvvw smvsehleec iitttkiikg 27961 neyifrvrav nkygigeple sdsvvaknaf vtpgppgipe vtkitknsmt vvwsrpiadg 28021 gsdisgyfle krdkkslgwf kvlketirdt rqkvtglten sdyqyrvcav naagqgpfse 28081 psefykaadp idppgppaki riadstkssi tlgwskpvyd ggsavtgyvv eirqgeeeew 28141 ttvstkgevr tteyvvsnlk pgvnyyfrvs avncagqgep iemnepvqak dileapeidl 28201 dvalrtsvia kagedvqvli pfkgrppptv twrkdeknlg sdarysient dssslltipq 28261 vtrndtgkyi ltiengvgep ksstvsvkvl dtpaacqklq vkhvsrgtvt llwdpplidg 28321 gspiinyvie krdatkrtws vvshkcssts fklidlsekt pfffrvlaen eigigepcet 28381 tepvkaaevp apirdlsmkd stktsvilsw tkpdfdggsv iteyvverkg kgeqtwshag 28441 isktceievs qlkeqsvlef rvfaknekgl sdpvtigpit vkeliitpev dlsdipgaqv 28501 tvrighnvhl elpykgkpkp siswlkdglp lkesefvrfs ktenkitlsi knakkehggk 28561 ytvildnavc riavpitvit lgppskpkgp irfdeikads vilswdvped ngggeitcys 28621 iekretsqtn wkmvcssvar ttfkvpnlvk daeyqfrvra enrygvsqpl vssiivakhq 28681 fripgppgkp viynvtsdgm sltwdapvyd ggsevtgfhv ekkernsilw qkvntspisg 28741 reyratglve gldyqfrvya ensaglssps dpskftlavs pvdppgtpdy idvtretitl 28801 kwnpplrdgg skivgysiek rqgnerwvrc nftdvsecqy tvtglspgdr yefriiarna 28861 vgtisppsqs sgiimtrden vppivefgpe yfdgliiksg eslrikalvq grpvprvtwf 28921 kdgveiekrm nmeitdvlgs tslfvrdatr dhrgvytvea knasgsakae ikvkvqdtpg 28981 kvvgpirftn itgekmtlww daplndgcap ithyiiekre tsrlawalie dkceaqsyta 29041 iklingneyq frvsavnkfg vgrpldsdpv vaqiqytvpd apgipepsni tgnsitltwa 29101 rpesdggsei qqyilerrek kstrwvkvis krpisetrfk vtgltegney efhvmaenaa 29161 gvgpasgisr likcrepvnp pgpptvvkvt dtskttvsle wskpvfdggm eiigyiiemc 29221 kadlgdwhkv naeacvktry tvtdlqagee ykfrvsaing agkgdscevt gtikavdrlt 29281 apeldidanf kqthvvraga sirlfiayqg rptptavwsk pdsnlslrad ihttdsfstl 29341 tvencnrnda gkytltvenn sgsksitftv kvldtpgppg pitfkdvtrg satlmwdapl 29401 ldggarihhy vvekreasrr swqvisekct rqifkvndla egvpyyfrvs avneygvgep 29461 yempepivat eqpapprrld vvdtskssav lawlkpdhdg gsritgylle mrqkgsdfwv 29521 eaghtkqltf tverlvekte yefrvkaknd agysepreaf ssviikepqi eptadltgit 29581 nqlitckags pftidvpisg rpapkvtwkl eemrlketdr vsitttkdrt tltvkdsmrg 29641 dsgryfltle ntagvktfsv tvvvigrpgp vtgpievssv saescvlswg epkdgggtei 29701 tnyivekres gttawqlvns svkrtqikvt hltkymeysf rvssenrfgv skplesapii 29761 aehpfvppsa ptrpevyhvs anamsirwee pyhdggskii gywvekkern tilwvkenkv 29821 pclecnykvt glvegleyqf rtyalnaagv skaseasrpi maqnpvdapg rpevtdvtrs 29881 tvsliwsapa ydggskvvgy iierkpvsev gdgrwlkcny tivsdnfftv talsegdtye 29941 frvlaknaag viskgsestg pvtcrdeyap pkaeldarlh gdlvtirags dlvldaavgg 30001 kpepkiiwtk gdkeldlcek vslqytgkra tavikfcdrs dsgkytltvk nasgtkavsv 30061 mvkvldspgp cgkltvsrvt qekctlawsl pqedggaeit hyiverrets rlnwvivege 30121 cptlsyvvtr liknneyifr vravnkygpg vpvesepiva rnsftipspp gipeevgtgk 30181 ehiiiqwtkp esdggneisn ylvdkrekks lrwtrvnkdy vvydtrlkvt slmegcdyqf 30241 rvtavnaagn sepseasnfi screpsytpg ppsaprvvdt tkhsislawt kpmydggtdi 30301 vgyvlemqek dtdqwyrvht natirnteft vpdlkmgqky sfrvaavnvk gmseysesia 30361 eiepveriei pdleladdlk ktvtiragas lrlmvsvsgr pppvitwskq gidlasraii 30421 dttesyslli vdkvnrydag kytieaenqs gkksatvlvk vydtpgpcps vkvkevsrds 30481 vtitweipti dggapvnnyi vekreaamra fktvttkcsk tlyrisglve gtmyyfrvlp 30541 eniygigepc etsdavlvse vplvpaklev vdvtkstvtl awekplydgg srltgyvlea 30601 ckagterwmk vvtlkptvle htvtslnege qylfriraqn ekgvsepret vtavtvqdlr 30661 vlptidlstm pqktihvpag rpvelvipia grpppaaswf fagsklrese rvtvethtkv 30721 akltiretti rdtgeytlel knvtgttset ikviildkpg pptgpikide idatsitisw 30781 eppeldggap lsgyvveqrd ahrpgwlpvs esvtrstfkf trltegneyv frvaatnrfg 30841 igsylqsevi ecrssiripg ppetlqifdv srdgmtltwy ppeddggsqv tgyiverkev 30901 radrwvrvnk vpvtmtryrs tgltegleye hrvtainarg sgkpsrpskp ivamdpiapp 30961 gkpqnprvtd ttrtsvslaw svpedeggsk vtgyliemqk vdqhewtkcn ttptkireyt 31021 lthlpqgaey rfrvlacnag gpgepaevpg tvkvtemley pdyelderyq egifvrqggv 31081 irltipikgk pfpickwtke gqdiskrami atsethtelv ikeadrgdsg tydlvlenkc 31141 gkkavyikvr vigspnspeg pleyddiqvr svrvswrppa ddggadilgy ilerrevpka 31201 awytidsrvr gtslvvkglk enveyhfrvs aenqfgiskp lkseepvtpk tplnppepps 31261 nppevldvtk ssvslswsrp kddggsrvtg yyierketst dkwvrhnktq itttmytvtg 31321 lvpdaeyqfr iiaqndvgls etspasepvv ckdpfdkpsq pgeleilsis kdsvtlqwek 31381 pecdggkeil gywveyrqsg dsawkksnke rikdkqftig glleateyef rvfaenetgl 31441 srprrtamsi ktkltsgeap girkemkdvt tklgeaaqls cqivgrplpd ikwyrfgkel 31501 iqsrkykmss dgrthtltvm teeqedegvy tciatnevge vetssklllq atpqfhpgyp 31561 lkekyygavg stlrlhvmyi grpvpamtwf hgqkllqnse nitientehy thlvmknvqr 31621 kthagkykvq lsnvfgtvda ildveiqdkp dkptgpivie allknsavis wkppaddggs 31681 witnyvvekc eakegaewql vssaisvttc rivnltenag yyfrvsaqnt fgisdplevs 31741 svviikspfe kpgapgkpti tavtkdscvv awkppasdgg akirnyylek rekkqnkwis 31801 vtteeiretv fsvknliegl eyefrvkcen lggesewsei sepitpksdv piqaphfkee 31861 lrnlnvryqs natlvckvtg hpkpivkwyr qgkeiiadgl kyriqefkgg yhqliiasvt 31921 dddatvyqvr atnqggsvsg taslevevpa kihlpktleg mgavhalrge vvsikipfsg 31981 kpdpvitwqk gqdlidnngh yqvivtrsft slvfpngver kdagfyvvca knrfgidqkt 32041 veldvadvpd pprgvkvsdv srdsvnltwt epasdggski tnyivekcat taerwlrvgq 32101 aretrytvin lfgktsyqfr viaenkfgls kpsepsepti tkedktramn ydeevdetre 32161 vsmtkashss tkelyekymi aedlgrgefg ivhrcvetss kktymakfvk vkgtdqvlvk 32221 keisilniar hrnilhlhes fesmeelvmi fefisgldif erintsafel nereivsyvh 32281 qvcealqflh shnighfdir peniiyqtrr sstikiiefg qarqlkpgdn frllftapey 32341 yapevhqhdv vstatdmwsl gtlvyvllsg inpflaetnq qiienimnae ytfdeeafke 32401 isieamdfvd rllvkerksr mtasealqhp wlkqkiervs tkvirtlkhr ryyhtlikkd 32461 lnmvvsaari scggairsqk gvsvakvkva sieigpvsgq imhavgeegg hvkyvckien 32521 ydqstqvtwy fgvrqlense kyeityedgv ailyvkditk lddgtyrckv vndygedssy 32581 aelfvkgvre vydyycrrtm kkikrrtdtm rllerppeft lplynktayv genvrfgvti 32641 tvhpephvtw yksgqkikpg dndkkytfes dkglyqltin svttdddaey tvvarnkyge 32701 dsckakltvt lhppptdstl rpmfkrllan aecqegqsvc feirvsgipp ptlkwekdgq 32761 plslgpniei ihegldyyal hirdtlpedt gyyrvtatnt agstscqahl qverlrykkq 32821 efkskeeher hvqkqidktl rmaeilsgte svpltqvake alreaavlyk pavstktvkg 32881 efrleieekk eerklrmpyd vpeprkykqt tieedqrikq fvpmsdmkwy kkirdqyemp 32941 gkldrvvqkr pkrirlsrwe qfyvmplpri tdqyrpkwri pklsqddlei vrparrrtps 33001 pdydfyyrpr rrslgdisde elllpiddyl amkrteeerl rleeelelgf sasppsrspp 33061 hfelsslrys spqahvkvee trkdfrysty hiptkaeast syaelrerha qaayrqpkqr 33121 qrimaerede ellrpvtttq hlseykseld fmskeeksrk ksrrqrevte iteieeeyei 33181 skhaqresss sasrllrrrr slsptyielm rpvselirsr pqpaeeyedd terrsptper 33241 trprspspvs serslsrfer sarfdifsry esmkaalktq ktserkyevl sqqpftldha 33301 pritlrmrsh rvpcgqntrf ilnvqskpta evkwyhngve lqesskihyt ntsgvltlei 33361 ldchtddsgt yravctnykg easdyatldv tggdyttyas qrrdeevprs vfpeltrtea 33421 yavssfkkts emeasssvre vksqmtetre slssyehsas aemksaalee ksleeksttr 33481 kikttlaari ltkprsmtvy egesarfscd tdgepvptvt wlrkgqvlst sarhqvtttk 33541 ykstfeissv qasdegnysv vvensegkqe aeftltiqka rvtekavtsp prvkspeprv 33601 kspeavkspk rvkspepshp kavsptetkp tptekvqhlp vsappkitqf lkaeaskeia 33661 kltcvvessv lrakevtwyk dgkklkengh fqfhysadgt yelkinnlte sdqgeyvcei 33721 sgeggtsktn lqfmgqafks ihekvskise tkksdqktte stvtrktepk apepisskpv 33781 ivtglqdttv ssdsvakfav katgeprpta iwtkdgkait qggkyklsed kggffleihk 33841 tdtsdsglyt ctvknsagsv sssckltika ikdteaqkvs tqktseitpq kkavvqeeis 33901 qkalrseeik mseaksqekl alkeeaskvl iseevkksaa tsleksivhe eitktsqase 33961 evrthaeika fstqmsineg qrlvlkania gatdvkwvln gveltnseey rygvsgsdqt 34021 ltikqashrd egiltciskt kegivkcqyd ltlskelsda pafisqprsq ninegqnvlf 34081 tceisgepsp eiewfknnlp isissnvsis rsrnvyslei rnasvsdsgk ytikaknfrg 34141 qcsataslmv lplveepsre vvlrtsgdts lqgsfssqsv qmsaskqeas fssfssssas 34201 smtemkfasm saqsmssmqe sfvemssssf mgisnmtqle sstskmlkag irgippkiea 34261 lpsdisideg kvltvacaft geptpevtws cggrkihsqe qgrfhientd dlttliimdv 34321 qkqdgglytl slgnefgsds atvnihirsi';

} else if (randomizer <= 100) {

peptidesequence = '1 mdqpqfsgap rfltrpkafv vsvgkdatls cqivgnptpq vswekdqqpv aagarfrlaq 61 dgdlyrltil dlalgdsgqy vcrarnaige afaavglqvd aeaacaeqap hfllrptsir 121 vregseatfr crvggsprpa vswskdgrrl gepdgprvrv eelgeasalr iraarprdgg 181 tyevraenpl gaasaaaalv vdsdaadtas rpgtstaall ahlqrrream raegapaspp 241 stgtrtctvt egkharlscy vtgepkpetv wkkdgqlvte grrhvvyeda qenfvlkilf 301 ckqsdrglyt ctasnlvgqt yssvlvvvre pavpfkkrlq dlevrekesa tflcevpqps 361 teaawfkeet rlwasakygi eeegterrlt vrnvsaddda vyicetpegs rtvaelavqg 421 nllrklprkt avrvgdtamf cvelavpvgp vhwlrnqeev vaggrvaisa egtrhtltis 481 qccledvgqv afmagdcqts tqfcvsaprk pplqppvdpv vkarmessvi lswsppphge 541 rpvtidgylv ekkklgtytw ircheaewva tpeltvadva eegnfqfrvs alnsfgqspy 601 lefpgtvhla pklavrtplk avqaveggev tfsvdltvas agewfldgqa lkassvyeih 661 cdrtrhtlti revpaslhga qlkfvangie ssirmevraa pgltankppa aaarevlarl 721 heeaqllael sdqaaavtwl kdgrtlspgp kyevqasagr rvllvrdvar ddaglyecvs 781 rggriayqls vqglarflhk dmagscvdav aggpaqfece tseahvhvhw ykdgmelghs 841 gerflqedvg trhrlvaatv trqdegtysc rvgedsvdfr lrvsepkvvf akeqlarrkl 901 qaeagasatl scevaqaqte vtwykdgkkl sssskvcmea tgctrrlvvq qagqadagey 961 sceaggqrls fhldvkepkv vfakdqvahs evqaeagasa tlscevaqaq tevmwykdgk 1021 klssslkvhv eakgcrrrlv vqqagktdag dysceargqr vsfrlhitep kmmfakeqsv 1081 hnevqaeaga samlscevaq aqtevtwykd gkklsssskv gmevkgctrr lvlpqagkad 1141 ageysceagg qrvsfhlhit epkgvfakeq svhnevqaea gttamlscev aqpqtevtwy 1201 kdgkklssss kvrmevkgct rrlvvqqvgk adageyscea ggqrvsfqlh itepkavfak 1261 eqlvhnevrt eagasatlsc evaqaqtevt wykdgkklss sskvrieaag cmrqlvvqqa 1321 gqadageytc eaggqrlsfh ldvsepkavf akeqlahrkv qaeagaiatl scevaqaqte 1381 vtwykdgkkl sssskvrmea vgctrrlvvq qacqadtgey sceaggqrls fsldvaepkv 1441 vfakeqpvhr evqaqagast tlscevaqaq tevmwykdgk klsfsskvrm eavgctrrlv 1501 vqqagqavag eysceagsqr lsfhlhvaep kavfakeqpa srevqaeagt satlscevaq 1561 aqtevtwykd gkklsssskv rmeavgctrr lvvqeagqad ageysckagd qrlsfhlhva 1621 epkvvfakeq pahrevqaea gasatlscev aqaqtevtwy kdgkklssss kvrveavgct 1681 rrlvvqqagq aeageyscea ggqqlsfrlq vaelepqise rpcrreplvv kehediilta 1741 tlatpsaatv twlkdgveir rskrhetasq gdthtltvhg aqvldsaiys crvgaegqdf 1801 pvqveevaak fcrllepvcg elggtvtlac elspacaevv wrcgntqlrv gkrfqmvaeg 1861 pvrsltvlgl raedageyvc esrddhtsaq ltvsvprvvk fmsglstvva eeggeatfqc 1921 vvspsdvavv wfrdgallqp sekfaisqsg ashsltisdl vledagqitv eaegasssaa 1981 lrvreapvlf kkklepqtve erssvtleve ltrpwpelrw trnatalapg knveihaega 2041 rhrlvlhnvg fadrgffgce tpddktqakl tvemrqvrlv rglqaveare qgtatmevql 2101 shadvdgswt rdglrfqqgp tchlavrgpm htltlsglrp edsglmvfka egvhtsarlv 2161 vtelpvsfsr plqdvvttek ekvtlecels rpnvdvrwlk dgvelragkt maiaaqgacr 2221 sltiyrcefa dqgvyvcdah daqssasvkv qgrtytliyr rvlaedagei qfvaenaesr 2281 aqlrvkelpv tlvrplrdki amekhrgvle cqvsrasaqv rwfkgsqelq pgpkyelvsd 2341 glyrkliisd vhaededtyt cdagdvktsa qffveeqsit ivrglqdvtv mepapawfec 2401 etsipsvrpp kwllgktvlq aggnvgleqe gtvhrlmlrr tcstmtgpvh ftvgksrssa 2461 rlvvsdipvv ltrplepktg relqsvvlsc dfrpapkavq wykddtplsp sekfkmsleg 2521 qmaelrilrl mpadagvyrc qagsahsste vtvearevtv tgplqdaeat eegwasfsce 2581 lshedeevew slngmplynd sfheishkgr rhtlvlksiq radagivras slkvstsarl 2641 evrvkpvvfl kalddlsaee rgtlalqcev sdpeahvvwr kdgvqlgpsd kydflhtagt 2701 rglvvhdvsp edaglytchv gseetrarvr vhdlhvgitk rlktmevleg escsfecvls 2761 hesasdpamw tvggktvgss srfqatrqgr kyilvvreaa psdagevvfs vrgltskasl 2821 ivrerpaaii kpledqwvap gedvelrcel sragtpvhwl kdrkairksq kydvvcegtm 2881 amlvirgasl kdageytcev easkstaslh veekancfte eltnlqveek gtavftckte 2941 hpaatvtwrk gllelrasgk hqpsqegltl rltisaleka dsdtytcdig qaqsraqllv 3001 qgrrvhiied ledvdvqegs satfrcrisp anyepvhwfl dktplhanel neidaqpggy 3061 hvltlrqlal kdsgtiyfea gdqrasaalr vtekpsvfsr eltdatiteg edltlvcets 3121 tcdipvcwtk dgktlrgsar cqlsheghra qllitgatlq dsgrykceag gacsssivrv 3181 harpvrfqea lkdlevlegg aatlrcvlss vaapvkwcyg nnvlrpgdky slrqegamle 3241 lvvrnlrpqd sgryscsfgd qttsatltvt alpaqfigkl rnkeategat atlrcelska 3301 apvewrkgse tlrdgdrycl rqdgamcelq irglamvdaa eyscvcgeer tsasltirpm 3361 pahfigrlrh qesiegatat lrcelskaap vewrkgresl rdgdrhslrq dgavcelqic 3421 glavadagey scvcgeerts atltvkalpa kfteglrnee avegatamlw celskvapve 3481 wrkgpenlrd gdryilrqeg trcelqicgl amadageylc vcgqertsat ltiralparf 3541 iedvknqear egatavlqce lnsaapvewr kgsetlrdgd ryslrqdgtk celqirglam 3601 adtgeyscvc gqertsamlt vralpikfte glrneeateg atavlrcels kmapvewwkg 3661 hetlrdgdrh slrqdgarce lqirglvaed ageylcmcgk ertsamltvr ampskfiegl 3721 rneeategdt atlwcelska apvewrkghe tlrdgdrhsl rqdgsrcelq irglavvdag 3781 eyscvcgqer tsatltvral parfiedvkn qearegatav lqcelskaap vewrkgsetl 3841 rggdryslrq dgtrcelqih glsvadtgey scvcgqerts atltvrapqp vfreplqslq 3901 aeegstatlq celseptatv vwskgglqlq angrreprlq gctaelvlqd lqredtgeyt 3961 ctcgsqatsa tltvtaapvr flrelqhqev deggtahlcc elsragasve wrkgslqlfp 4021 cakyqmvqdg aaaellvrgv eqedagdytc dtghtqsmas lsvrvprpkf ktrlqsleqe 4081 tgdiarlccq lsdaesgavv qwlkegvelh agpkyemrsq gatrellihq leakdtgeya 4141 cvtggqktaa slrvtepevt ivrglvdaev tadedvefsc evsragatgv qwclqglplq 4201 snevtevavr dgrihtlrlk gvtpedagtv sfhlgnhass aqltvrapev tileplqdvq 4261 lsegqdasfq crlsrasgqe arwalggvpl qanemnditv eqgtlhlltl hkvtledagt 4321 vsfhvgtcss eaqlkvtakn tvvrglenve aleggealfe cqlsqpevaa htwllddepv 4381 htsenaevvf fenglrhlll lknlrpqdsc rvtflagdmv tsafltvrgw rleileplkn 4441 aavragaqac ftctlseavp vgeaswying aavqpddsdw tvtadgshha lllrsaqphh 4501 agevtfacrd avasarltvl glpdppedae vvarsshtvt lswaapmsdg ggglcgyrve 4561 vkegatgqwr lchelvpgpe cvvdglapge tyrfrvaavg pvgagepvhl pqtvrlaepp 4621 kpvppqpsap esrqvaaged vslelevvae ageviwhkgm eriqpggrfe vvsqgrqqml 4681 vikgftaedq geyhcglaqg sicpaaatfq valspasvde apqpslppea aqegdlhllw 4741 ealarkrrms reptldsise lpeedgrsqr lpqeaeevap dlsegystad elartgdadl 4801 shtssddesr agtpslvtyl kkagrpgtsp laskvgapaa psvkpqqqqe plaavrpplg 4861 dlstkdlgdp smdkaavkiq aafkgykvrk emkqqegpmf shtfgdteaq vgdalrlecv 4921 vaskadvrar wlkdgveltd grhhhidqlg dgtcsllitg ldradagcyt cqvsnkfgqv 4981 thsacvvvsg seseaesssg gelddafrra arrlhrlfrt kspaevsdee lflsadegpa 5041 epeepadwqt yredehfici rfealtearq avtrfqemfa tlgigveikl veqgprrvem 5101 cisketpapv vppeplpsll tsdaapvflt elqnqevqdg ypvsfdcvvt gqpmpsvrwf 5161 kdgklleedd hyminedqqg ghqliitavv padmgvyrcl aensmgvsst kaelrvdlts 5221 tdydtaadat esssyfsaqg ylssreqegt esttdegqlp qvveelrdlq vapgtrlakf 5281 qlkvkgypap rlywfkdgqp ltasahirmt dkkilhtlei isvtredsgq yaayisnamg 5341 aayssarllv rgpdepeekp asdvheqlvp prmlerftpk kvkkgssitf svkvegrpvp 5401 tvhwlreeae rgvlwigpdt pgytvassaq qhslvlldvg rqhqgtytci asnaagqalc 5461 saslhvsglp kveeqekvke alistflqgt tqaisaqgle tasfadlggq rkeeplaake 5521 alghlslaev gteeflqklt sqitemvsak itqaklqvpg gdsdedsktp sasprhgrsr 5581 psssiqesss esedgdarge ifdiyvvtad ylplgaeqda itlregqyve vldaahplrw 5641 lvrtkptkss psrqgwvspa yldrrlklsp ewgaaeapef pgeavsedey karlssviqe 5701 llsseqafve elqflqshhl qhlercphvp iavagqkavi frnvrdigrf hssflqelqq 5761 cdtdddvamc fiknqaafeq yleflvgrvq aesvvvstai qefykkyaee allagdpsqp 5821 pppplqhyle qpvervqryq allkelirnk arnrqncall eqayavvsal pqraenklhv 5881 slmenypgtl qalgepirqg hfivwegapg armpwkghnr hvflfrnhlv ickprrdsrt 5941 dtvsyvfrnm mklssidlnd qvegddrafe vwqeredsvr kyllqartai iksswvkeic 6001 giqqrlalpv wrppdfeeel adctaelget vklacrvtgt pkpviswykd gkavqvdphh 6061 iliedpdgsc alildsltgv dsgqymcfaa saagncstlg kilvqvpprf vnkvraspfv 6121 egedaqftct iegapypqir wykdgalltt gnkfqtlsep rsgllvlvir aaskedlgly 6181 ecelvnrlgs arasaelriq spmlqaqeqc hreqlvaave vteqetkvpk ktviieetit 6241 tvvksprgqr rspskspsrs psrcsasplr pgllapdlly lpgagqprrp eaepgqkpvv 6301 ptlyvteaea hspalpglsg pqpkwvevee tievrvkkmg pqgvspttev prsssghlft 6361 lpgatpggdp nsnnsnnkll aqeawaqgta mvgvreplvf rvdargsvdw aasgmgslee 6421 egtmeeagee egedgdafvt eesqdthslg drdpkilthn grmltladle dyvpgegetf 6481 hcggpgpgap ddppcevsvi qreigeptvg qpvllsvgha lgprgplglf rpeprgaspp 6541 gpqvrslegt sfllreapar pvgsapwtqs fctrirrsad sgqssfttel stqtvnfgtv 6601 getvtlhicp drdgdeaaqp';

} else if (randomizer <= 150) {

peptidesequence = ' 1 mlwweevedc yeredvqkkt ftkwvnaqfs kfgkqhienl fsdlqdgrrl ldllegltgq 61 klpkekgstr vhalnnvnka lrvlqnnnvd lvnigstdiv dgnhkltlgl iwniilhwqv 121 knvmknimag lqqtnsekil lswvrqstrn ypqvnvinft tswsdglaln alihshrpdl 181 fdwnsvvcqq satqrlehaf niaryqlgie klldpedvdt typdkksilm yitslfqvlp 241 qqvsieaiqe vemlprppkv tkeehfqlhh qmhysqqitv slaqgyerts spkprfksya 301 ytqaayvtts dptrspfpsq hleapedksf gsslmesevn ldryqtalee vlswllsaed 361 tlqaqgeisn dvevvkdqfh thegymmdlt ahqgrvgnil qlgskligtg klsedeetev 421 qeqmnllnsr weclrvasme kqsnlhrvlm dlqnqklkel ndwltkteer trkmeeeplg 481 pdledlkrqv qqhkvlqedl eqeqvrvnsl thmvvvvdes sgdhataale eqlkvlgdrw 541 anicrwtedr wvllqdillk wqrlteeqcl fsawlseked avnkihttgf kdqnemlssl 601 qklavlkadl ekkkqsmgkl yslkqdllst lknksvtqkt eawldnfarc wdnlvqklek 661 staqisqavt ttqpsltqtt vmetvttvtt reqilvkhaq eelpppppqk krqitvdsei 721 rkrldvdite lhswitrsea vlqspefaif rkegnfsdlk ekvnaierek aekfrklqda 781 srsaqalveq mvnegvnads ikqaseqlns rwiefcqlls erlnwleyqn niiafynqlq 841 qleqmtttae nwlkiqpttp septaiksql kickdevnrl sdlqpqierl kiqsialkek 901 gqgpmfldad fvaftnhfkq vfsdvqarek elqtifdtlp pmryqetmsa irtwvqqset 961 klsipqlsvt dyeimeqrlg elqalqsslq eqqsglyyls ttvkemskka pseisrkyqs 1021 efeeiegrwk klssqlvehc qkleeqmnkl rkiqnhiqtl kkwmaevdvf lkeewpalgd 1081 seilkkqlkq crllvsdiqt iqpslnsvne ggqkikneae pefasrlete lkelntqwdh 1141 mcqqvyarke alkgglektv slqkdlsemh ewmtqaeeey lerdfeyktp delqkaveem 1201 krakeeaqqk eakvklltes vnsviaqapp vaqealkkel etlttnyqwl ctrlngkckt 1261 leevwacwhe llsylekank wlnevefklk ttenipggae eisevldsle nlmrhsednp 1321 nqirilaqtl tdggvmdeli neeletfnsr wrelheeavr rqklleqsiq saqetekslh 1381 liqesltfid kqlaayiadk vdaaqmpqea qkiqsdltsh eisleemkkh nqgkeaaqrv 1441 lsqidvaqkk lqdvsmkfrl fqkpanfeqr lqeskmilde vkmhlpalet ksveqevvqs 1501 qlnhcvnlyk slsevkseve mviktgrqiv qkkqtenpke ldervtalkl hynelgakvt 1561 erkqqlekcl klsrkmrkem nvltewlaat dmeltkrsav egmpsnldse vawgkatqke 1621 iekqkvhlks itevgealkt vlgkketlve dklsllnsnw iavtsraeew lnllleyqkh 1681 metfdqnvdh itkwiiqadt lldesekkkp qqkedvlkrl kaelndirpk vdstrdqaan 1741 lmanrgdhcr klvepqisel nhrfaaishr iktgkasipl keleqfnsdi qkllepleae 1801 iqqgvnlkee dfnkdmnedn egtvkellqr gdnlqqritd erkreeikik qqllqtkhna 1861 lkdlrsqrrk kaleishqwy qykrqaddll kclddiekkl aslpeprder kikeidrelq 1921 kkkeelnavr rqaeglsedg aamaveptqi qlskrwreie skfaqfrrln faqihtvree 1981 tmmvmtedmp leisyvpsty lteithvsqa lleveqllna pdlcakdfed lfkqeeslkn 2041 ikdslqqssg ridiihskkt aalqsatpve rvklqealsq ldfqwekvnk mykdrqgrfd 2101 rsvekwrrfh ydikifnqwl teaeqflrkt qipenwehak ykwylkelqd gigqrqtvvr 2161 tlnatgeeii qqssktdasi lqeklgslnl rwqevckqls drkkrleeqk nilsefqrdl 2221 nefvlwleea dniasiplep gkeqqlkekl eqvkllveel plrqgilkql netggpvlvs 2281 apispeeqdk lenklkqtnl qwikvsralp ekqgeieaqi kdlgqlekkl edleeqlnhl 2341 llwlspirnq leiynqpnqe gpfdvketei avqakqpdve eilskgqhly kekpatqpvk 2401 rkledlssew kavnrllqel rakqpdlapg lttigasptq tvtlvtqpvv tketaiskle 2461 mpsslmlevp aladfnrawt eltdwlslld qviksqrvmv gdledinemi ikqkatmqdl 2521 eqrrpqleel itaaqnlknk tsnqeartii tdrieriqnq wdevqehlqn rrqqlnemlk 2581 dstqwleake eaeqvlgqar akleswkegp ytvdaiqkki tetkqlakdl rqwqtnvdva 2641 ndlalkllrd ysaddtrkvh miteninasw rsihkrvser eaaleethrl lqqfpldlek 2701 flawlteaet tanvlqdatr kerlledskg vkelmkqwqd lqgeieahtd vyhnldensq 2761 kilrslegsd davllqrrld nmnfkwselr kkslnirshl eassdqwkrl hlslqellvw 2821 lqlkddelsr qapiggdfpa vqkqndvhra fkrelktkep vimstletvr iflteqpleg 2881 leklyqepre lppeeraqnv trllrkqaee vnteweklnl hsadwqrkid etlerlrelq 2941 eatdeldlkl rqaevikgsw qpvgdllids lqdhlekvka lrgeiaplke nvshvndlar 3001 qlttlgiqls pynlstledl ntrwkllqva vedrvrqlhe ahrdfgpasq hflstsvqgp 3061 weraispnkv pyyinhetqt tcwdhpkmte lyqsladlnn vrfsayrtam klrrlqkalc 3121 ldllslsaac daldqhnlkq ndqpmdilqi inclttiydr leqehnnlvn vplcvdmcln 3181 wllnvydtgr tgrirvlsfk tgiislckah ledkyrylfk qvasstgfcd qrrlglllhd 3241 siqiprqlge vasfggsnie psvrscfqfa nnkpeieaal fldwmrlepq smvwlpvlhr 3301 vaaaetakhq akcnickecp iigfryrslk hfnydicqsc ffsgrvakgh kmhypmveyc 3361 tpttsgedvr dfakvlknkf rtkryfakhp rmgylpvqtv legdnmetpv tlinfwpvds 3421 apasspqlsh ddthsriehy asrlaemens ngsylndsis pnesiddehl liqhycqsln 3481 qdsplsqprs paqilisles eergeleril adleeenrnl qaeydrlkqq hehkglsplp 3541 sppemmptsp qsprdaelia eakllrqhkg rlearmqile dhnkqlesql hrlrqlleqp 3601 qaeakvngtt vsspstslqr sdssqpmllr vvgsqtsdsm geedllsppq dtstgleevm 3661 eqlnnsfpss rgrntpgkpm redtm';

} else {

peptidesequence = '1 maddedyeev veyyteevvy eevpgetitk iyettttrts dyeqsetskp alaqpalaqp 61 asakpverrk virkkvdpsk fmtpyiahsq kmqdlfspnk ykekfektkg qpyasttdtp 121 elrrikkvqd qlsevkyrmd gdvaktichv dekakdieha kkvsqqvskv lykqnwedtk 181 dkyllppdap elvqavknta mfskklyted weadkslfyp yndspelrrv aqaqkalsdv 241 aykkglaeqq aqftpladpp diefakkvtn qvskqkyked yenkikgkws etpcfevana 301 rmnadnistr kyqedfenmk dqiyfmqtet peykmnkkag vaaskvkyke dyeknkgkad 361 ynvlpasenp qlrqlkaagd alsdklyken yektkaksin ycetpkfkld tvlqnfssdk 421 kykdsylkdi lghyvgsfed pyhshcmkvt aqnsdknyka eyeedrgkgf fpqtitqeye 481 aikkldqckd htykvhpdkt kftqvtdspv llqaqvnskq lsdlnykakh esekfkchip 541 pdtpafiqhk vnaynlsdnl ykqdwekska kkfdikvdai pllaakantk ntsdvmykkd 601 yeknkgkmig vlsinddpkm lhslkvaknq sdrlykenye ktkaksmnyc etpkyqldtq 661 lknfsearyk dlyvkdvlgh yvgsmedpyh thcmkvaaqn sdksykaeye edkgkcyfpq 721 titqeyeaik kldqckdhty kvhpdktkft avtdspvllq aqlntkqlsd lnykakhege 781 kfkchipada pqfiqhrvna ynlsdnvykq dwekskakkf dikvdaipll aakantknts 841 dvmykkdyek skgkmigals inddpkmlhs lktaknqsdr eyrkdyeksk tiytapldml 901 qvtqakksqa iasdvdykhi lhsysyppds invdlakkay alqsdveyka dynswmkgcg 961 wvpfgsleme kakrasdiln ekkyrqhpdt lkftsiedap itvqskinqa qrsdiaykak 1021 geeiihkynl ppdlpqfiqa kvnaynisen mykadlkdls kkgydlrtda ipiraakaar 1081 qaasdvqykk dyekakgkmv gfqslqddpk lvhymnvaki qsdreykkdy ektkskyntp 1141 hdmfnvvaak kaqdvvsnvn ykhslhhyty lpdamdlels knmmqiqsdn vykedynnwm 1201 kgigwipigs ldvekvkkag dalnekkyrq hpdtlkftsi vdspvmvqak qntkqvsdil 1261 ykakgedvkh kytmspdlpq flqakcnayn isdvcykrdw ydliakgnnv lgdaipitaa 1321 kasrniasdy kykeayeksk gkhvgfrslq ddpklvhymn vaklqsdrey kknyentkts 1381 yhtpgdmvsi taakmaqdva tnvnykqplh hytylpdams lehtrnvnqi qsdnvykdey 1441 nsflkgigwi pigslevekv kkagdalner kyrqhpdtvk ftsvpdsmgm vlaqhntkql 1501 sdlnykvege klkhkytidp elpqfiqakv nalnmsdahy kadwkktiak gydlrpdaip 1561 ivaakssrni asdckykeay ekakgkqvgf lslqddpklv hymnvakiqs dreykkgyea 1621 sktkyhtpld mvsvtaakks qevatnanyr qsyhhytllp dalnvehsrn amqiqsdnly 1681 ksdftnwmkg igwvpiesle vekakkagei lsekkyrqhp eklkftyamd tmeqalnksn 1741 klnmdkrlyt ekwnkdktti hvmpdtpdil lsrvnqitms dklykagwee ekkkgydlrp 1801 daiaikaara srdiasdyky kkayeqakgk higfrsledd pklvhfmqva kmqsdreykk 1861 gyeksktsfh tpvdmlsvva akksqevatn anyrnvihty nmlpdamsfe laknmmqiqs 1921 dnqykadyad fmkgigwlpl gsleaeknkk ameiisekky rqhpdtlkys tlmdsmnmvl 1981 aqnnakimne hlykqawead ktkvhimpdi pqiilakana inmsdklykl sleeskkkgy 2041 dlrpdaipik aakasrdias dykykynyek gkgkmvgfrs leddpklvhs mqvakmqsdr 2101 eykknyentk tsyhtpadml svtaakdaqa nitntnykhl ihkyillpda mnieltrnmn 2161 riqsdneykq dynewykglg wspagsleve kakkateyas dqkyrqhpsn fqfkkltdsm 2221 dmvlakqnah tmnkhlytid wnkdktkihv mpdtpdilqa kqnqtlysqk lyklgweeal 2281 kkgydlpvda isvqlakasr diasdykykq gyrkqlghhv gfrslqddpk lvlsmnvakm 2341 qsereykkdf ekwktkfssp vdmlgvvlak kcqelvsdvd yknylhqwtc lpdqndvvqa 2401 kkvyelqsen lyksdlewlr gigwsplgsl eaeknkrase iisekkyrqp pdrnkftsip 2461 damdivlakt naknrsdrly reawdkdktq ihimpdtpdi vlakanlint sdklyrmgye 2521 elkrkgydlp vdaipikaak asreiaseyk ykegfrkqlg hhigarnied dpkmmwsmhv 2581 akiqsdreyk kdfekwktkf sspvdmlgvv lakkcqtlvs dvdyknylhq wtclpdqsdv 2641 iharqaydlq sdnlyksdlq wlkgigwmts gsledeknkr atqilsdhvy rqhpdqfkfs 2701 slmdsipmvl aknnaitmnh rlyteawdkd kttvhimpdt pevllakqnk vnyseklykl 2761 gleeakrkgy dmrvdaipik aakasrdias efkykegyrk qlghhigara irddpkmmws 2821 mhvakiqsdr eykkdfekwk tkfsspvdml gvvlakkcqt lvsdvdykny lhqwtclpdq 2881 sdviharqay dlqsdnmyks dlqwmrgigw vsigsldvek ckrateilsd kiyrqppdrf 2941 kftsvtdsle qvlaknnait mnkrlyteaw dkdktqihim pdtpeimlar mnkinysesl 3001 yklaneeakk kgydlrsdai pivaakasrd iisdykykdg yckqlghhig arnieddpkm 3061 mwsmhvakiq sdreykkdfe kwktkfsspv dmlgvvlakk cqtlvsdvdy knylhewtcl 3121 pdqsdvihar qaydlqsdni yksdlqwlrg igwvpigsmd vvkckratei lsdniyrqpp 3181 dklkftsvtd sleqvlaknn alnmnkrlyt eawdkdktqi himpdtpeim larqnkinys 3241 etlyklanee akkkgydlrs daipivaaka srdvisdyky kdgyrkqlgh higarniedd 3301 pkmmwsmhva kiqsdreykk dfekwktkfs spvdmlgvvl akkcqtlvsd vdyknylhew 3361 tclpdqndvi harqaydlqs dniyksdlqw lrgigwvpig smdvvkckra aeilsdniyr 3421 qppdklkfts vtdsleqvla knnalnmnkr lyteawdkdk tqvhimpdtp eimlarqnki 3481 nyseslyrqa meeakkegyd lrsdaipiva akasrdiasd ykykeayrkq lghhigarav 3541 hddpkimwsl hiakvqsdre ykkdfekykt rysspvdmlg ivlakkcqtl vsdvdykhpl 3601 hewiclpdqn diiharkayd lqsdnlyksd lewmkgigwv pidslevvra kragellsdt 3661 iyrqrpetlk ftsitdtpeq vlaknnalnm nkrlyteawd ndkktihvmp dtpeimlakl 3721 nrinysdkly klaleeskke gydlrldaip iqaakasrdi asdykykegy rkqlghhiga 3781 rnikddpkmm wsihvakiqs dreykkefek wktkfsspvd mlgvvlakkc qilvsdidyk 3841 hplhewtclp dqndviqark aydlqsdaiy ksdlewlrgi gwvpigsvev ekvkrageil 3901 sdrkyrqpad qlkftcitdt peivlaknna ltmskhlyte awdadktsih vmpdtpdill 3961 aksnsanisq klytkgwdes kmkdydlrad aisiksakas rdiasdykyk eayekqkghh 4021 igaqsieddp kimcaihagk iqsereykke fqkwktkfss pvdmlsilla kkcqtlvtdi 4081 dyrnylhewt cmpdqndiiq akkaydlqsd svykadlewl rgigwmpegs vemnrvkvaq 4141 dlvnerlyrt rpealsftsi vdtpevvlak anslqisekl yqeawnkdks nitipsdtpe 4201 mlqahinalq isnklyqkdw ndakqkgydi radaieikha kasreiasey kykegyrkql 4261 ghhmgfrtlq ddpksvwaih aakiqsdrey kkayekskgi hntpldmmsi vqakkcqvlv 4321 sdidyrnylh qwtclpdqnd viqakkaydl qsdnlyksdl ewlkgigwlp egsvevmrvk 4381 naqnllnerl yrikpealkf tsivdtpevi qakinavqis eplyrdawek ekanvnvpad 4441 tplmlqskin alqisnkryq qawedvkmtg ydlradaigi qhakasrdia sdylyktaye 4501 kqkghyigcr sakedpklvw aanvlkmqnd rlykkayndh kakisipvdm vsisaakegq 4561 alasdvdyrh ylhhwscfpd qndviqarka ydlqsdsvyk adlewlrgig wmpegsvemn 4621 rvkvaqdlvn erlyrtrpea lsftsivdtp evvlakansl qiseklyqea wnkdksniti 4681 psdtpemlqa hinalqisnk lyqkdwndtk qkgydirada ieikhakasr eiaseykyke 4741 gyrkqlghhm gfrtlqddpk svwaihaaki qsdreykkay ekskgihntp ldmmsivqak 4801 kcqvlvsdid yrnylhqwtc lpdqndviqa kkaydlqsdn lyksdlewlk gigwlpegsv 4861 evmrvknaqn llnerlyrik pealkftsiv dtpeviqaki navqiseply rnawekekan 4921 vnvpadtplm lqskinalqi snkryqqawe dvkmtgydlr adaigiqhak asrdiasdyl 4981 yktayekqkg hyigcrsake dpklvwaanv lkmqndrlyk kayndhkaki sipvdmvsis 5041 aakegqalas dvdyrhylhh wscfpdqndv iqarkaydlq sdsvykadle wlrgigwmpe 5101 gsvemnrvkv aqdlvnerly rtrpealsft sivdtpevvl akanslqise klyqeawnkd 5161 ksnitipsdt pemlqahina lqisnklyqk dwndtkqkgy diradaieik hakasreias 5221 eykykegyrk qlghhmgfrt lqddpksvwa ihaakiqsdr eykkayeksk gihntpldmm 5281 sivqakkcqv lvsdidyrny lhqwtclpdq ndviqakkay dlqsdnlyks dlewlkgigw 5341 lpegsvevmr vknaqnllne rlyrikpeal kftsivdtpe viqakinavq iseplyrdaw 5401 ekekanvnvp adtplmlqsk inalqisnkr yqqawedvkm tgydlradai giqhakasrd 5461 iasdylykta yekqkghyig crsakedpkl vwaanvlkmq ndrlykkayn dhkakisipv 5521 dmvsisaake gqalasdvdy rhylhrwscf pdqndviqar kaydlqsdal ykadlewlrg 5581 igwmpqgspe vlrvknaqni fcdsvyrtpv vnlkytsivd tpevvlaksn aenisipkyr 5641 evwdkdktsi himpdtpein laranalnvs nklyregwde mkagcdvrld aipiqaakas 5701 reiasdykyk ldhekqkghy vgtltarddn kirwaliadk lqnereyrld wakwkakiqs 5761 pvdmlsilhs knsqalvsdm dyrnylhqwt cmpdqndviq akkayelqsd nvykadlewl 5821 rgigwmpnds vsvnhakhaa difsekkyrt kietlnftpv ddrvdyvtak qsgeilddik 5881 yrkdwnatks kytltetpll htaqeaaril dqylykegwe rqkatgyilp pdavpfvhah 5941 hcndvqselk ykaehvkqkg hyvgvptmrd dpklvwfeha gqiqnerlyk edyhktkaki 6001 nipadmvsvl aakqgqtlvs didyrnylhq wmchpdqndv iqarkaydlq sdnvyradle 6061 wlrgigwipl dsvdhvrvtk nqemmsqiky kknalenypn frsvvdppei vlakinsvnq 6121 sdvkyketfn kakgkytfsp dtphishskd mgklystily kgawegtkay gytlderyip 6181 ivgakhadlv nselkykety ekqkghylag kvigefpgvv hcldfqkmrs alnyrkhyed 6241 tkanvhipnd mmnhvlakrc qyilsdleyr hyfhqwtsll eepnvirvrn aqeilsdnvy 6301 kddlnwlkgi gcyvwdtpqi lhakksydlq sqlqytaagk enlqnynlvt dtplyvtavq 6361 sginasevky kenyhqikdk yttvletvdy drtrnlknly ssnlykeawd rvkatsyilp 6421 sstlslthak nqkhlashik yreeyekfka lytlprsvdd dpntarclrv gklnidrlyr 6481 svyeknkmki hivpdmvemv takdsqkkvs eidyrlrlhe wichpdlqvn dhvrkvtdqi 6541 sdivykddln wlkgigcyvw dtpeilhakh aydlrddiky kahmlktrnd yklvtdtpvy 6601 vqavksgkql sdavyhydyv hsvrgkvapt tktvdldral hayklqssnl yktslrtlpt 6661 gyrlpgdtph fkhikdtrym ssyfkykeay ehtkaygytl gpkdvpfvhv rrvnnvtser 6721 lyrelyhklk dkihttpdtp eirqvkktqe avseliyksd ffkmqghmis lpytpqvihc 6781 ryvgditsdi kykedlqvlk gfgcflydtp dmvrsrhlrk lwsnylytdk arkmrdkykv 6841 vldtpeyrkv qelkthlsel vyraagkkqk siftsvpdtp dllrakrgqk lqsqylyvel 6901 atkerphhha gnqttalkha kdvkdmvsek kykiqyekmk dkytpvpdtp ilirakrayw 6961 nasdlryket fqktkgkyht vkdaldivyh rkvtddiski kykenymsql giwrsipdrp 7021 ehfhhravtd tvsdvkyked ltwlkgigcy aydtpdftla eknktlysky kykevfertk 7081 sdfkyvadsp inrhfkyatq lmnekkyrad yeqrkdkyhl vvdeprhlla ktagdqisqi 7141 kyrknyeksk dkftsivdtp ehlrttkvnk qisdilykle ynkakprgyt tihdtpmllh 7201 vrkvkdevsd lkykevyqrn ksnctiepda vhikaakday kvntnldykk qyeankahwk 7261 wtpdrpdflq aaksslqqsd feykldrefl kgcklsvtdd kntvlalrnt liesdlkyke 7321 khvkergtch avpdtpqill aktvsnlvse nkykdhvkkh laqgsyttlp etrdtvhvke 7381 vtkhvsdtny kkkfvkekgk snysimlepp evkhamevak kqsdvayrkd akenlhyttv 7441 adrpdikkat qaakqaseve yrakhrkegs hglsmlgrpd iemakkaakl ssqvkyrenf 7501 dkekgktpky npkdsqlykv mkdannlase vkykadlkkl hkpvtdmkes limnhvlnts 7561 qlassyqykk kyekskghyh tipdnleqlh lkeatelqsi vkykekyeke rgkpmldfet 7621 ptyitakesq qmqsgkeyrk dyeesikgrn ltglevtpal lhvkyatkia sekeyrkdle 7681 esirgkglte medtpdmlra knatqilnek eykrdlelev kgrglnaman etpdfmrarn 7741 atdiasqiky kqsaemekan ftsvvdtpei ihaqqvknls sqkkykedae ksmsyyetvl 7801 dtpeiqrvre nqknfsllqy qcdlknskgk itvvqdtpei lrvkenqknf ssvlykedvs 7861 pgtaigktpe mmrvkqtqdh issvkykeai gqgtpipdlp evkrvketqk hissvmyken 7921 lgtgipttvt peiervkrnq enfssvlyke nlgkgiptpi tpemervkrn qenfssilyk 7981 enlskgtplp vtpemervkl nqenfssvly kenvgkgipi pitpemervk hnqenfssvl 8041 ykenlgtgip ipitpemqrv khnqenlssv lykenmgkgt plpvtpemer vkhnqeniss 8101 vlykenmgkg tplpvtpeme rvkhnqenis svlykenmgk gtplavtpem ervkhnqeni 8161 ssvlykenvg katatpvtpe mqrvkrnqen issvlykenl gkatptpftp emervkrnqe 8221 nfssvlyken mrkatptpvt pemerakrnq enissvlysd sfrkqiqgka ayvldtpemr 8281 rvretqrhis tvkyhedfek hkgcftpvvt dpitervkkn mqdfsdinyr giqrkvveme 8341 qkrndqdqet itglrvwrtn pgsvfdydpa edniqsrslh minvqaqrrs reqsrsasal 8401 sisggeekse hseapdhhls tysdggvfav staykhaktt elpqqrsssv atqqttvssi 8461 pshpstagki framydymaa dadevsfkdg daiinvqaid egwmygtvqr tgrtgmlpan 8521 yveai';

}

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.732619659e+29 - 4.950341882e+28);

cursor2x = spawnerx - (1.23758547e+29 - 9.28189103e+27);

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 9.28189103e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

} else if (randomizer <= 50) {

peptidesequence = '1 mviqkekksc gqvveewkef vwnprthqfm grtgtswafi llfylvfygf ltamftltmw 61 vmlqtvsdht pkyqdrlatp glmirpkten ldvivnvsdt eswdqhvqkl nkflepynds 121 iqaqkndvcr pgryyeqpdn gvlnypkrac qfnrtqlgnc sgigdsthyg ystgqpcvfi 181 kmnrvinfya ganqsmnvtc agkrdedaen lgnfvmfpan gnidlmyfpy ygkkfhvnyt 241 qplvavkfln vtpnvevnve crinaaniat dderdkfagr vafklrinkt';

} else if (randomizer <= 75) {

peptidesequence = '1 mtknekksln qslaewklfi ynpttgeflg rtakswglil lfylvfygfl aalfsftmwv 61 mlqtlndevp kyrdqipspg lmvfpkpvta leytfsrsdp tsyagyiedl kkflkpytle 121 eqknltvcpd galfeqkgpv yvacqfpisl lqacsgmndp dfgysqgnpc ilvkmnriig 181 lkpegvprid cvsknedipn vavyphngmi dlkyfpyygk klhvgylqpl vavqvsfapn 241 ntgkevtvec kidgsanlks qddrdkflgr vmfkitara';

} else {

peptidesequence = ' 1 mrrqlrsrra psfpysyryr lddpdeanqn yladeeeeae eearvtvvpk seeeeeeeek 61 eeeeeeekee eegqgqptgn awwqklqims eylwdperrm flartgqsws lilliyfffy 121 aslaavitlc mytlfltisp yiptftervk ppgvmirpfa hslnfnfnvs epdtwqhyvi 181 slngflqgyn dslqeemnvd cppgqyfiqd gnededkkac qfkrsflknc sgledptfgy 241 stgqpcillk mnrivgfrpe lgdpvkvsck vqrgdendir sisyypesas fdlryypyyg 301 klthvnytsp lvamhftdvv knqavpvqcq lkgkgvindv indrfvgrvi ftlniet';

}

protein();

} else if (randomizer <= 75) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 medghsktve qslnffgtdp ergltldqik anqkkygpne lpteegksiw qlvleqfddl 61 lvkilllaai isfvlalfee heetftafve plvillilia navvgvwqer naesaiealk 121 eyepemgkvv rqdksgiqkv rakeivpgdl vevsvgdkip adirithiys ttlridqsil 181 tgesvsvikh tdaipdprav nqdkknilfs gtnvaagkar gvvigtglst aigkirtems 241 eteeiktplq qkldefgeql skvisvicva vwainighfn dpahggswik gaiyyfkiav 301 alavaaipeg lpavittcla lgtrrmakkn aivrslpsve tlgctsvics dktgtlttnq 361 msvsrmfifd kvegndssfl efemtgstye pigevflngq rikaadydtl qelsticimc 421 ndsaidynef kqafekvgea tetalivlae klnsfsvnks gldrrsaaia crgeietkwk 481 keftlefsrd rksmssyctp lkasrlgtgp klfvkgapeg vlerctharv gttkvpltsa 541 lkakilaltg qygtgrdtlr clalavadsp mkpdemdlgd stkfyqyevn ltfvgvvgml 601 dpprkevfds ivrcraagir vivitgdnka taeaicrrig vfaededttg ksysgrefdd 661 lspteqkaav arsrlfsrve pqhkskivef lqsmneisam tgdgvndapa lkkaeigiam 721 gsgtavaksa aemvladdnf ssivsaveeg raiynnmkqf irylissnig evvsifltaa 781 lglpealipv qllwvnlvtd glpatalgfn ppdldimekp prkadeglis gwlffrymai 841 gfyvgaatvg aaawwfvfsd egpklsywql thhlsclggg defkgvdcki fsdphamtma 901 lsvlvtieml namnslsenq slitmppwcn lwligsmals ftlhfvilyv dvlstvfqvt 961 plsaeewitv mkfsipvvll detlkfvark iadvpdvvvd rm';

protein();

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mgevtaeeve kfldsnigfa kqyynlhyra klisdllgak eaavdfsnyh spssmeesei 61 ifdllrdfqe nlqtekcifn vmkklcfllq adrmslfmyr trngiaelat rlfnvhkdav 121 ledclvmpdq eivfpldmgi vghvahskki anvpnteede hfcdfvdilt eyktknilas 181 pimngkdvva iimavnkvdg shftkrdeei llkylnfanl imkvyhlsyl hncetrrgqi 241 llwsgskvfe eltdierqfh kalytvrafl ncdrysvgll dmtkqkeffd vwpvlmgevp 301 pysgprtpdg reinfykvid yilhgkedik vipnpppdhw alvsglpayv aqnglicnim 361 napaedffaf qkepldesgw miknvlsmpi vnkkeeivgv atfynrkdgk pfdemdetlm 421 esltqflgws vlnpdtyesm nklenrkdif qdivkyhvkc dneeiqkilk trevygkepw 481 eceeeelaei lqaelpdadk yeinkfhfsd lpltelelvk cgiqmyyelk vvdkfhipqe 541 alvrfmysls kgyrkityhn wrhgfnvgqt mfsllvtgkl kryftdleal amvtaafchd 601 idhrgtnnly qmksqnplak lhgssilerh hlefgktllr deslnifqnl nrrqhehaih 661 mmdiaiiatd lalyfkkrtm fqkivdqskt yeseqewtqy mmleqtrkei vmammmtacd 721 lsaitkpwev qsqvallvaa efweqgdler tvlqqnpipm mdrnkadelp klqvgfidfv 781 ctfvykefsr fheeitpmld gitnnrkewk aladeydakm kvqeekkqkq qsaksaaagn 841 qpggnpspgg attskscciq';

} else if (randomizer <= 40) {

peptidesequence = '1 mslseeqars fldqnpdfar qyfgkklspe nvaaacedgc ppdcdslrdl cqveestall 61 elvqdmqesi nmervvfkvl rrlctllqad rcslfmyrqr ngvaelatrl fsvqpdsvle 121 dclvppdsei vfpldigvvg hvaqtkkmvn vedvaecphf ssfadeltdy ktknmlatpi 181 mngkdvvavi mavnklngpf ftsededvfl kylnfatlyl kiyhlsylhn cetrrgqvll 241 wsankvfeel tdierqfhka fytvraylnc erysvglldm tkekeffdvw svlmgesqpy 301 sgprtpdgre ivfykvidyv lhgkeeikvi ptpsadhwal asglpsyvae sgficnimna 361 sademfkfqe galddsgwli knvlsmpivn kkeeivgvat fynrkdgkpf deqdevlmes 421 ltqflgwsvm ntdtydkmnk lenrkdiaqd mvlyhvkcdr deiqlilptr arlgkepadc 481 dedelgeilk eelpgpttfd iyefhfsdle cteldlvkcg iqmyyelgvv rkfqipqevl 541 vrflfsiskg yrrityhnwr hgfnvaqtmf tllmtgklks yytdleafam vtaglchdid 601 hrgtnnlyqm ksqnplaklh gssilerhhl efgkfllsee tlniyqnlnr rqhehvihlm 661 diaiiatdla lyfkkramfq kivdesknyq dkkswveyls lettrkeivm ammmtacdls 721 aitkpwevqs kvallvaaef weqgdlertv ldqqpipmmd rnkaaelpkl qvgfidfvct 781 fvykefsrfh eeilpmfdrl qnnrkewkal adeyeakvka leekeeeerv aakkvgteic 841 nggpapksst ccil';

} else if (randomizer <= 60) {

peptidesequence = '1 mgeinqvave kyleenpqfa keyfdrklrv evlgeifkns qvpvqssmsf seltqveesa 61 lclellwtvq eeggtpeqgv hralqrlahl lqadrcsmfl crsrngipev asrlldvtpt 121 skfednlvgp dkevvfpldi givgwaahtk kthnvpdvkk nshfsdfmdk qtgyvtknll 181 atpivvgkev lavimavnkv nasefskqde evfskylnfv siilrlhhts ymyniesrrs 241 qilmwsankv feeltdverq fhkalytvrs ylncerysig lldmtkekef ydewpiklge 301 vepykgpktp dgrevnfyki idyilhgkee ikviptppad hwtlisglpt yvaengficn 361 mmnapadeyf tfqkgpvdet gwviknvlsl pivnkkediv gvatfynrkd gkpfdehdey 421 itetltqflg wsllntdtyd kmnklenrkd iaqemlmnqt katpeeiksi lkfqeklnvd 481 viddceekql vailkedlpd prsaelyefr fsdfpltehg likcgirlff einvvekfkv 541 pvevltrwmy tvrkgyravt yhnwrhgfnv gqtmftllmt grlkkyytdl eafamlaaaf 601 chdidhrgtn nlyqmkstsp larlhgssil erhhleyskt llqdeslnif qnlnkrqfet 661 vihlfevaii atdlalyfkk rtmfqkivda ceqmqteeea ikyvtvdptk keiimammmt 721 acdlsaitkp wevqsqvalm vanefweqgd lertvlqqqp ipmmdrnkrd elpklqvgfi 781 dfvctfvyke fsrfhkeitp mlsglqnnrv ewksladeyd akmkvieeea kkqeggaeka 841 aedsgggddk ksktclml';

} else if (randomizer <= 80) {

peptidesequence = '1 msdnttlpap asnqgpttpr kgppkfkqrq trqfkskppk kgvkgfgddi pgmeglgtdi 61 tvicpweafs hlelhelaqf gii';

} else if (randomizer <= 100) {

peptidesequence = '1 mnleppkaef rsatrvaggp vtprkgppkf kqrqtrqfks kppkkgvqgf gddipgmegl 61 gtditvicpw eafnhlelhe laqygii';

}

protein();

}

} else {

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.732619659e+29 + 4.950341882e+28);

cursor2x = spawnerx - (1.23758547e+29 - 4.950341882e+28);

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 4.950341882e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

randomize\_sarcomere\_protein();

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.732619659e+29 + 4.950341882e+28);

cursor2x = spawnerx - (1.23758547e+29 - 4.950341882e+28);

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 1.172612233e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 150);

if (randomizer <= 50) {

peptidesequence = ' 10 20 30 40 50 MGLSDGEWQL VLNVWGKVEA DIPGHGQEVL IRLFKGHPET LEKFDKFKHL 60 70 80 90 100 KSEDEMKASE DLKKHGATVL TALGGILKKK GHHEAEIKPL AQSHATKHKI 110 120 130 140 150 PVKYLEFISE CIIQVLQSKH PGDFGADAQG AMNKALELFR KDMASNYKEL GFQG ';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mrmsvglsll lplsgrtfll llsvvmaqsh wpsepseavr dwenqleasm hsvlsdlhea 61 vptvvgipdg tavvgrsfrv tiptdliass gdiikvsaag kealpswlhw dsqshtlegl 121 pldtdkgvhy isvsatrlga ngshipqtss vfsievyped hselqsvrta spdpgevvss 181 acaadepvtv ltvildadlt kmtpkqridl lhrmrsfsev elhnmklvpv vnnrlfdmsa 241 fmagpgnakk vvengallsw klgcslnqns vpdihgveap aregamsaql gypvvgwhia 301 nkkpplpkrv rrqihatptp vtaigpptta iqeppsrivp tptspaiapp tetmappvrd 361 pvpgkptvti rtrgaiiqtp tlgpiqptrv seagttvpgq irptmtipgy veptavatpp 421 ttttkkprvs tpkpatpstd stttttrrpt kkprtprpvp rvttkvsitr letaspptri 481 rtttsgvprg gepnqrpelk nhidrvdawv gtyfevkips dtfydhedtt tdklkltlkl 541 reqqlvgeks wvqfnsnsql myglpdsshv gkheyfmhat dkgglsavda feihvhrrpq 601 gdraparfka kfvgdpalvl ndihkkialv kklafafgdr ncstitlqni trgsivvewt 661 nntlplepcp keqiaglsrr iaeddgkprp afsnalepdf katsitvtgs gscrhlqfip 721 vvpprrvpse apptevpdrd peksseddvy lhtvipavvv aailliagii amicyrkkrk 781 gkltledqat fikkgvpiif adelddskpp psssmplilq eekaplpppe ypnqsvpett 841 plnqdtmgey tplrdedpna ppyqppppft apmegkgsrp knmtpyrspp pyvpp';

protein();

} else {

heme();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1.123758547e+29);

cursor1y = spawnery - 1 \* 6.187927353e+26;

cursor1x = spawnerx - (1.175706197e+30 - 1.23758547e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((37.5 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1.123758547e+29);

cursor1y = spawnery - 1 \* 6.187927353e+26;

cursor1x = spawnerx - (1.175706197e+30 - 2.475170941e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((37.5 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1.123758547e+29);

cursor1y = spawnery - 1 \* 6.187927353e+26;

cursor1x = spawnerx - (1.175706197e+30 - 3.712756412e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((37.5 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1.123758547e+29);

cursor1y = spawnery - 1 \* 6.187927353e+26;

cursor1x = spawnerx - (1.175706197e+30 - 4.950341882e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((37.5 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

}

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

set\_earth\_rotation();

if (cursor1x > spawnerx - 1.175706197e+30) {

randomize\_cytosol();

} else if (cursor1z < spawnerz + 1.243773398e+29 && cursor1y > spawnery - 1.243773398e+29) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cells\_list.push(['myocyte', spawnerz, spawnery, spawnerx, spawnerz + 1.23758546e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function neuromuscular\_junction() {

cursor2z = spawnerz + 6.187927353e+26;

cursor2y = spawnery - 6.187927353e+26;

cursor2x = spawnerx - 1.175706197e+30;

for (countcell1 = 0; countcell1 < 3; countcell1++) {

if (unused\_variable == 5) {

lipid1 = 3.093963676e+28;

lipid2 = 1.856378206e+28;

lipid\_wall\_z();

cursor2z = cursor2z + 1.231397543e+29;

lipid\_wall\_z();

cursor2z = cursor2z - 1.231397543e+29;

lipid2 = 1.231397543e+29;

lipid\_wall\_y();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 6.187927353e+26;

cursor1z = cursor2z + 6.187927353e+26;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6.806720089e+25 \* 100) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.187927353e+26) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mepwpllllf slcsaglvlg sehetrlvak lfkdyssvvr pvedhrqvve vtvglqliql 61 invdevnqiv ttnvrlkqgd mvdlprpscv tlgvplfshl qneqwvdynl kwnpddyggv 121 kkihipseki wrpdlvlynn adgdfaivkf tkvllqytgh itwtppaifk syceiivthf 181 pfdeqncsmk lgtwtydgsv vainpesdqp dlsnfmesge wvikesrgwk hsvtysccpd 241 tpyldityhf vmqrlplyfi vnviipcllf sfltglvfyl ptdsgekmtl sisvllsltv 301 fllvivelip stssavplig kymlftmvfv iasiiitviv inthhrspst hvmpnwvrkv 361 fidtipnimf fstmkrpsre kqdkkifted idisdisgkp gpppmgfhsp likhpevksa 421 iegikyiaet mksdqesnna aaewkyvamv mdhillgvfm lvciigtlav fagrlielnq 481 qg';

} else if (randomizer <= 40) {

peptidesequence = '1 mtpgallmll galgaplapg vrgseaegrl reklfsgyds svrparevgd rvrvsvglil 61 aqlislnekd eemstkvyld lewtdyrlsw dpaehdgids lritaesvwl pdvvllnnnd 121 gnfdvaldis vvvssdgsvr wqppgiyrss csiqvtyfpf dwqnctmvfs sysydssevs 181 lqtglgpdgq ghqeihiheg tfiengqwei ihkpsrliqp pgdprggreg qrqevifyli 241 irrkplfylv nviapcilit llaifvfylp pdagekmgls ifalltltvf lllladkvpe 301 tslsvpiiik ylmftmvlvt fsvilsvvvl nlhhrsphth qmplwvrqif ihklplylrl 361 krpkperdlm pepphcsspg sgwgrgtdey firkppsdfl fpkpnrfqpe lsapdlrrfi 421 dgpnravall pelrevvssi syiarqlqeq edhdalkedw qfvamvvdrl flwtfiifts 481 vgtlviflda tyhlpppdpf p';

} else if (randomizer <= 60) {

peptidesequence = ' 1 megpvltlgl laalavcgsw glneeerlir hlfqekgynk elrpvahkee svdvalaltl 61 snlislkeve etlttnvwie hgwtdnrlkw naeefgnisv lrlppdmvwl peivlennnd 121 gsfqisyscn vlvyhygfvy wlppaifrss cpisvtyfpf dwqncslkfs slkytakeit 181 lslkqdaken rtypvewiii dpegftenge weivhrparv nvdpraplds psrqditfyl 241 iirrkplfyi inilvpcvli sfmvnlvfyl padsgektsv aisvllaqsv fllliskrlp 301 atsmaiplig kfllfgmvlv tmvvvicviv lnihfrtpst hvlsegvkkl fletlpellh 361 msrpaedgps pgalvrrsss lgyiskaeey fllksrsdlm fekqserhgl arrlttarrp 421 passeqaqqe lfnelkpavd ganfivnhmr dqnnyneekd swnrvartvd rlclfvvtpv 481 mvvgtawifl qgvynqpppq pfpgdpysyn vqdkrfi';

} else if (randomizer <= 80) {

peptidesequence = '1 maraplgvll llgllgrgvg kneelrlyhh lfnnydpgsr pvrepedtvt islkvtltnl 61 islnekeetl ttsvwigidw qdyrlnyskd dfggietlrv pselvwlpei vlennidgqf 121 gvaydanvlv yeggsvtwlp paiyrsvcav evtyfpfdwq ncslifrsqt ynaeeveftf 181 avdndgktin kididteayt engewaidfc pgvirrhhgg atdgpgetdv iysliirrkp 241 lfyviniivp cvlisglvll ayflpaqagg qkctvsinvl laqtvflfli aqkipetsls 301 vpllgrflif vmvvatlivm ncvivlnvsq rtptthamsp rlrhvllell prllgspppp 361 eapraasppr rassvglllr aeelilkkpr selvfegqrh rqgtwtaafc qslgaaapev 421 rccvdavnfv aestrdqeat geevsdwvrm gnaldnicfw aalvlfsvgs sliflgayfn 481 rvpdlpyapc iqp';

} else {

peptidesequence = '1 mhggqgplll llllavclga qgrnqeerll adlmqnydpn lrpaerdsdv vnvslkltlt 61 nlislneree alttnvwiem qwcdyrlrwd prdyeglwvl rvpstmvwrp divlennvdg 121 vfevalycnv lvspdgciyw lppaifrsac sisvtyfpfd wqncslifqs qtystneidl 181 qlsqedgqti ewifidpeaf tengewaiqh rpakmlldpa apaqeaghqk vvfylliqrk 241 plfyviniia pcvlissvai lihflpakag gqkctvainv llaqtvflfl vakkvpetsq 301 avpliskylt fllvvtiliv vnavvvlnvs lrsphthsma rgvrkvflrl lpqllrmhvr 361 plapaavqdt qsrlqngssg wsittgeeva lclprsellf qqwqrqglva aaleklekgp 421 elglsqfcgs lkqaapaiqa cveacnliac arhqqshfdn gneewflvgr vldrvcflam 481 lslficgtag iflmahynrv palpfpgdpr pylpspd';

}

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 1.856378206e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.856378206e+28;

lipid1 = 1.856378206e+28;

cursor2x = cursor2x - 3.093963676e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 3.093963676e+28;

cursor2y = cursor2y - 1.856378206e+28;

lipid\_wall\_x();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((6.806720089e+25 \* 15.4) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.187927353e+26) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 150);

if (randomizer <= 50) {

peptidesequence = ' 10 20 30 40 50 MRPPQCLLHT PSLASPLLLL LLWLLGGGVG AEGREDAELL VTVRGGRLRG 60 70 80 90 100 IRLKTPGGPV SAFLGIPFAE PPMGPRRFLP PEPKQPWSGV VDATTFQSVC 110 120 130 140 150 YQYVDTLYPG FEGTEMWNPN RELSEDCLYL NVWTPYPRPT SPTPVLVWIY 160 170 180 190 200 GGGFYSGASS LDVYDGRFLV QAERTVLVSM NYRVGAFGFL ALPGSREAPG 210 220 230 240 250 NVGLLDQRLA LQWVQENVAA FGGDPTSVTL FGESAGAASV GMHLLSPPSR 260 270 280 290 300 GLFHRAVLQS GAPNGPWATV GMGEARRRAT QLAHLVGCPP GGTGGNDTEL 310 320 330 340 350 VACLRTRPAQ VLVNHEWHVL PQESVFRFSF VPVVDGDFLS DTPEALINAG 360 370 380 390 400 DFHGLQVLVG VVKDEGSYFL VYGAPGFSKD NESLISRAEF LAGVRVGVPQ 410 420 430 440 450 VSDLAAEAVV LHYTDWLHPE DPARLREALS DVVGDHNVVC PVAQLAGRLA 460 470 480 490 500 AQGARVYAYV FEHRASTLSW PLWMGVPHGY EIEFIFGIPL DPSRNYTAEE 510 520 530 540 550 KIFAQRLMRY WANFARTGDP NEPRDPKAPQ WPPYTAGAQQ YVSLDLRPLE 560 570 580 590 600 VRRGLRAQAC AFWNRFLPKL LSATDTLDEA ERQWKAEFHR WSSYMVHWKN 610 QFDHYSKQDR CSDL ';

} else if (randomizer <= 100) {

peptidesequence = '1 mlwweevedc yeredvqkkt ftkwvnaqfs kfgkqhienl fsdlqdgrrl ldllegltgq 61 klpkekgstr vhalnnvnka lrvlqnnnvd lvnigstdiv dgnhkltlgl iwniilhwqv 121 knvmknimag lqqtnsekil lswvrqstrn ypqvnvinft tswsdglaln alihshrpdl 181 fdwnsvvcqq satqrlehaf niaryqlgie klldpedvdt typdkksilm yitslfqvlp 241 qqvsieaiqe vemlprppkv tkeehfqlhh qmhysqqitv slaqgyerts spkprfksya 301 ytqaayvtts dptrspfpsq hleapedksf gsslmesevn ldryqtalee vlswllsaed 361 tlqaqgeisn dvevvkdqfh thegymmdlt ahqgrvgnil qlgskligtg klsedeetev 421 qeqmnllnsr weclrvasme kqsnlhrvlm dlqnqklkel ndwltkteer trkmeeeplg 481 pdledlkrqv qqhkvlqedl eqeqvrvnsl thmvvvvdes sgdhataale eqlkvlgdrw 541 anicrwtedr wvllqdillk wqrlteeqcl fsawlseked avnkihttgf kdqnemlssl 601 qklavlkadl ekkkqsmgkl yslkqdllst lknksvtqkt eawldnfarc wdnlvqklek 661 staqisqavt ttqpsltqtt vmetvttvtt reqilvkhaq eelpppppqk krqitvdsei 721 rkrldvdite lhswitrsea vlqspefaif rkegnfsdlk ekvnaierek aekfrklqda 781 srsaqalveq mvnegvnads ikqaseqlns rwiefcqlls erlnwleyqn niiafynqlq 841 qleqmtttae nwlkiqpttp septaiksql kickdevnrl sdlqpqierl kiqsialkek 901 gqgpmfldad fvaftnhfkq vfsdvqarek elqtifdtlp pmryqetmsa irtwvqqset 961 klsipqlsvt dyeimeqrlg elqalqsslq eqqsglyyls ttvkemskka pseisrkyqs 1021 efeeiegrwk klssqlvehc qkleeqmnkl rkiqnhiqtl kkwmaevdvf lkeewpalgd 1081 seilkkqlkq crllvsdiqt iqpslnsvne ggqkikneae pefasrlete lkelntqwdh 1141 mcqqvyarke alkgglektv slqkdlsemh ewmtqaeeey lerdfeyktp delqkaveem 1201 krakeeaqqk eakvklltes vnsviaqapp vaqealkkel etlttnyqwl ctrlngkckt 1261 leevwacwhe llsylekank wlnevefklk ttenipggae eisevldsle nlmrhsednp 1321 nqirilaqtl tdggvmdeli neeletfnsr wrelheeavr rqklleqsiq saqetekslh 1381 liqesltfid kqlaayiadk vdaaqmpqea qkiqsdltsh eisleemkkh nqgkeaaqrv 1441 lsqidvaqkk lqdvsmkfrl fqkpanfeqr lqeskmilde vkmhlpalet ksveqevvqs 1501 qlnhcvnlyk slsevkseve mviktgrqiv qkkqtenpke ldervtalkl hynelgakvt 1561 erkqqlekcl klsrkmrkem nvltewlaat dmeltkrsav egmpsnldse vawgkatqke 1621 iekqkvhlks itevgealkt vlgkketlve dklsllnsnw iavtsraeew lnllleyqkh 1681 metfdqnvdh itkwiiqadt lldesekkkp qqkedvlkrl kaelndirpk vdstrdqaan 1741 lmanrgdhcr klvepqisel nhrfaaishr iktgkasipl keleqfnsdi qkllepleae 1801 iqqgvnlkee dfnkdmnedn egtvkellqr gdnlqqritd erkreeikik qqllqtkhna 1861 lkdlrsqrrk kaleishqwy qykrqaddll kclddiekkl aslpeprder kikeidrelq 1921 kkkeelnavr rqaeglsedg aamaveptqi qlskrwreie skfaqfrrln faqihtvree 1981 tmmvmtedmp leisyvpsty lteithvsqa lleveqllna pdlcakdfed lfkqeeslkn 2041 ikdslqqssg ridiihskkt aalqsatpve rvklqealsq ldfqwekvnk mykdrqgrfd 2101 rsvekwrrfh ydikifnqwl teaeqflrkt qipenwehak ykwylkelqd gigqrqtvvr 2161 tlnatgeeii qqssktdasi lqeklgslnl rwqevckqls drkkrleeqk nilsefqrdl 2221 nefvlwleea dniasiplep gkeqqlkekl eqvkllveel plrqgilkql netggpvlvs 2281 apispeeqdk lenklkqtnl qwikvsralp ekqgeieaqi kdlgqlekkl edleeqlnhl 2341 llwlspirnq leiynqpnqe gpfdvketei avqakqpdve eilskgqhly kekpatqpvk 2401 rkledlssew kavnrllqel rakqpdlapg lttigasptq tvtlvtqpvv tketaiskle 2461 mpsslmlevp aladfnrawt eltdwlslld qviksqrvmv gdledinemi ikqkatmqdl 2521 eqrrpqleel itaaqnlknk tsnqeartii tdrieriqnq wdevqehlqn rrqqlnemlk 2581 dstqwleake eaeqvlgqar akleswkegp ytvdaiqkki tetkqlakdl rqwqtnvdva 2641 ndlalkllrd ysaddtrkvh miteninasw rsihkrvser eaaleethrl lqqfpldlek 2701 flawlteaet tanvlqdatr kerlledskg vkelmkqwqd lqgeieahtd vyhnldensq 2761 kilrslegsd davllqrrld nmnfkwselr kkslnirshl eassdqwkrl hlslqellvw 2821 lqlkddelsr qapiggdfpa vqkqndvhra fkrelktkep vimstletvr iflteqpleg 2881 leklyqepre lppeeraqnv trllrkqaee vnteweklnl hsadwqrkid etlerlrelq 2941 eatdeldlkl rqaevikgsw qpvgdllids lqdhlekvka lrgeiaplke nvshvndlar 3001 qlttlgiqls pynlstledl ntrwkllqva vedrvrqlhe ahrdfgpasq hflstsvqgp 3061 weraispnkv pyyinhetqt tcwdhpkmte lyqsladlnn vrfsayrtam klrrlqkalc 3121 ldllslsaac daldqhnlkq ndqpmdilqi inclttiydr leqehnnlvn vplcvdmcln 3181 wllnvydtgr tgrirvlsfk tgiislckah ledkyrylfk qvasstgfcd qrrlglllhd 3241 siqiprqlge vasfggsnie psvrscfqfa nnkpeieaal fldwmrlepq smvwlpvlhr 3301 vaaaetakhq akcnickecp iigfryrslk hfnydicqsc ffsgrvakgh kmhypmveyc 3361 tpttsgedvr dfakvlknkf rtkryfakhp rmgylpvqtv legdnmetpv tlinfwpvds 3421 apasspqlsh ddthsriehy asrlaemens ngsylndsis pnesiddehl liqhycqsln 3481 qdsplsqprs paqilisles eergeleril adleeenrnl qaeydrlkqq hehkglsplp 3541 sppemmptsp qsprdaelia eakllrqhkg rlearmqile dhnkqlesql hrlrqlleqp 3601 qaeakvngtt vsspstslqr sdssqpmllr vvgsqtsdsm geedllsppq dtstgleevm 3661 eqlnnsfpss rgrntpgkpm redtm';

} else {

peptidesequence = ' 1 mgqdqtkqqi ekglqlyqsn qtekalqvwt kvlekssdlm grfrvlgclv tahsemgryk 61 emlkfavvqi dtareledad fllesylnla rsneklcefh ktisycktcl glpgtragaq 121 lggqvslsmg naflglsvfq kalesfekal ryahnnddam lecrvccslg sfyaqvkdye 181 kalffpckaa elvnnygkgw slkyramsqy hmavayrllg rlgsamecce esmkialqhg 241 drplqalcll cfadihrsrg dletafpryd samsimteig nrlgqvqall gvakcwvark 301 aldkaldaie raqdlaeevg nklsqlklhc lsesiyrskg lqrelrahvv rfhecveete 361 lycglcgesi geknsrlqal pcshifhlrc lqnngtrscp ncrrssmkpg fv';

}

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 1.856378206e+28;

}

}

lipid1 = 1.107638996e+29;

lipid\_wall\_x();

cursor2z = spawnerz + 6.187927353e+26;

cursor2y = spawnery - 6.187927353e+26;

cursor2x = spawnerx - 1.208502212e+30;

lipid2 = 1.23758547e+29;

lipid1 = 2.908325856e+28 - (4.085906752e+25 + 2);

lipid\_wall\_y();

cursor2y = cursor2y - 1.23758547e+29;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+29;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+29;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+29;

lipid1 = 1.23758547e+29;

lipid2 = 1.23758547e+29;

lipid\_wall\_x();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 6.187927353e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1.23758547e+29 \* 1) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.23758547e+29) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_synaptic\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 6.187927353e+27;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

for (countsub1 = 0; countsub1 < Math.round((1.175706197e+29 \* 1) / 4.950341882e+27 - 0); countsub1++) {

for (countsub2 = 0; countsub2 < Math.round((1 \* 1.23758547e+28) / 4.950341882e+27 - 0); countsub2++) {

for (countsub3 = 0; countsub3 < Math.round((1 \* 1.175706197e+29) / 4.950341882e+27 - 0); countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

synaptic\_vesicle\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function tendon\_1() {

lipid2 = 1.23758547e+30 - 1.23758547e+27;

lipid1 = 1.175706197e+30 - 6.187927353e+26;

cursor2z = spawnerz + 10 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 6.187927353e+26;

cursor2x = spawnerx - 1 \* 6.187927353e+26;

lipid\_wall\_z();

lipid1 = 1.23758547e+30 - 1.23758547e+27;

lipid2 = 9.28189103e+29 - (4.085906752e+25 + 1);

lipid\_wall\_y();

cursor2y = spawnery - (1.23758547e+30 - 6.187927353e+26);

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 6.187927353e+26;

lipid1 = 1.175706197e+30 - 6.187927353e+26;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 1.175706197e+30;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 6.187927353e+26;

if (unused\_variable == 5) {

cursor1z = cursor2z;

cursor1y = cursor2y;

cursor1x = cursor2x;

density\_parameter\_x = 1.856378206e+26 \* 1;

density\_parameter\_y = 1.856378206e+26 \* 1;

density\_parameter\_z = 1.856378206e+26;

generation\_parameter\_x = Math.round((1 \* 1.175706197e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(8.663098295e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1;

cursor1y = spawnery + 1;

cursor1x = spawnerx + 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 1.175706197e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(3.093963676e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1;

cursor1y = spawnery + 1;

cursor1x = spawnerx + 1.175706197e+30;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 6.12604808e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1.175706197e+30 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_cytosol();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

list\_cells.push(['tendon 1', spawnerz, spawnery, spawnerx, spawnerz + 1.23758546e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function tendon\_2() {

lipid2 = 1.23758547e+30 - 1.23758547e+27;

lipid1 = 1.175706197e+30 - 6.187927353e+26;

cursor2z = spawnerz + 30 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 6.187927353e+26;

cursor2x = spawnerx - 1 \* 6.187927353e+26;

lipid\_wall\_z();

cursor2z = spawnerz + 1 \* 1;

lipid1 = 1.23758547e+30 - 1.23758547e+27;

lipid2 = 9.28189103e+29 - (4.085906752e+25 + 1);

lipid\_wall\_y();

cursor2y = spawnery - (1.23758547e+30 - 6.187927353e+26);

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 6.187927353e+26;

lipid1 = 1.175706197e+30 - 6.187927353e+26;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 1.175706197e+30;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 6.187927353e+26;

if (unused\_variable == 5) {

cursor1z = cursor2z;

cursor1y = cursor2y;

cursor1x = cursor2x;

density\_parameter\_x = 1.856378206e+26 \* 1;

density\_parameter\_y = 1.856378206e+26 \* 1;

density\_parameter\_z = 1.856378206e+26;

generation\_parameter\_x = Math.round((1 \* 1.175706197e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(8.663098295e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 9.28189103e+29;

cursor1y = spawnery + 1;

cursor1x = spawnerx + 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 1.175706197e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(2.475170941e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1;

cursor1y = spawnery + 1;

cursor1x = spawnerx + 1.175706197e+30;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 6.12604808e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.175706197e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(1.175706197e+30 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 10);

generation\_parameter\_z = Math.round((1.23758547e+30 + 0) / density\_parameter\_x - 10);

}

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_cytosol();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

list\_cells.push(['tendon 1', spawnerz, spawnery, spawnerx, spawnerz + 1.23758546e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function cardiomyocyte\_4() {

default\_cardiomyocyte\_1();

default\_cardiomyocyte\_2();

cells\_list.push(['cardiomyocyte 4', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function cardiomyocyte\_1() {

default\_cardiomyocyte\_1();

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 6.187927353e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 5;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.806720089e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 5;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cardiomyocyte\_2();

cells\_list.push(['cardiomyocyte 1', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function cardiomyocyte\_3() {

default\_cardiomyocyte\_1();

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 6.187927353e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 1;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.806720089e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 1;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cardiomyocyte\_2();

cells\_list.push(['cardiomyocyte 3', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function cardiomyocyte\_2() {

default\_cardiomyocyte\_1();

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 6.187927353e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 3;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.806720089e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 1.652176603e+28 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 3;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 msgsksvspp gyaaqktaap aprggpehrs awgeadsran gyphapggsa rgstkkpgga 61 vtpqqqqrla srwrsddddd pplsgddpla ggfgfsfrsk sawqerggdd cgrgsrrqrr 121 gaasggstra ppagggggsa aaaasaggte vrprsvevgl eerrgkgraa deleagaveg 181 gegsgdggss adsgsgagpg avlslgaccl allqifrskk fpsdklerly qryffrlnqs 241 sltmlmavlv lvclvmlafh aarpplqlpy lavlaaavgv ilimavlcnr aafhqdhmgl 301 acyaliavvl avqvvglllp qprsasegiw wtvffiytiy tllpvrmraa vlsgvllsal 361 hlaialrtna qdqfllkqlv snvlifsctn ivgvcthypa evsqrqafqe treciqarlh 421 sqrenqqqer lllsvlprhv amemkadina kqedmmfhki yiqkhdnvsi lfadiegfts 481 lasqctaqel vmtlnelfar fdklaaenhc lrikilgdcy ycvsglpear adhahccvem 541 gmdmieaisl vrevtgvnvn mrvgihsgrv hcgvlglrkw qfdvwsndvt lanhmeaggk 601 agrihitkat lnylngdyev epgcggerna ylkehsietf lilrctqkrk eekamiakmn 661 rqrtnsighn pphwgaerpf ynhlggnqvs kemkrmgfed pkdknaqesa npedevdefl 721 graidarsid rlrsehvrkf lltfrepdle kkyskqvddr fgayvacasl vflficfvqi 781 tivphsifml sfyltcslll tlvvfvsviy scvklfpspl qtlsrkivrs kmnstlvgvf 841 titlvflaaf vnmftcnsrd llgclaqehn isasqvnach vaesavnysl gdeqgfcgsp 901 wpncnfpeyf tysvllslla csvflqisci gklvlmlaie liyvlivevp gvtlfdnadl 961 lvtanaidff nngtsqcpeh atkvalkvvt piiisvfvla lylhaqqves tarldflwkl 1021 qateekeeme elqaynrrll hnilpkdvaa hflarerrnd elyyqscecv avmfasianf 1081 sefyvelean negveclrll neiiadfdei isedrfrqle kiktigstym aasglndsty 1141 dkvgkthika ladfamklmd qmkyinehsf nnfqmkigln igpvvagvig arkpqydiwg 1201 ntvnvasrmd stgvpdriqv ttdmyqvlaa ntyqlecrgv vkvkgkgemm tyflnggppl 1261 s';

protein();

} else {

peptidesequence = ' 1 mdklppsmrk rlyslpqqvg akawimdeee daeeegaggr qdpsrrsirl rplpspspsa 61 aaggtesrss algaadsegp argagksstn gdcrrfrgsl aslgsrgggs ggtgsgsshg 121 hlhdsaeerr liaegdaspg edrtppglaa eperpgasaq paasppppqq ppqpasasce 181 qpsvdtaikv eggaaagdqi lpeaevrlgq agfmqrqfga mlqpgvnkfs lrmfgsqkav 241 ereqervksa gfwiihpysd frfywdltml llmvgnliii pvgitffkde nttpwivfnv 301 vsdtfflidl vlnfrtgivv ednteiildp qrikmkylks wfmvdfissi pvdyiflive 361 tridsevykt aralrivrft kilsllrllr lsrliryihq weeifhmtyd lasavvrivn 421 ligmmlllch wdgclqflvp mlqdfpddcw vsinnmvnns wgkqysyalf kamshmlcig 481 ygrqapvgms dvwltmlsmi vgatcyamfi ghataliqsl dssrrqyqek ykqveqymsf 541 hklppdtrqr ihdyyehryq gkmfdeesil gelseplree iinfncrklv asmplfanad 601 pnfvtsmltk lrfevfqpgd yiiregtigk kmyfiqhgvv svltkgnket kladgsyfge 661 iclltrgrrt asvradtycr lyslsvdnfn evleeypmmr rafetvaldr ldrigkknsi 721 llhkvqhdln sgvfnyqene iiqqivqhdr emahcahrvq aaasatptpt pviwtpliqa 781 plqaaaatts vaialthhpr lpaaifrppp gsglgnlgag qtprhlkrlq slipsalgsa 841 spasspsqvd tpssssfhiq qlagfsapag lspllpssss spppgacgsp saptpsagva 901 attiagfghf hkalggslss sdsplltplq pgarspqaaq pspappgarg glglpehflp 961 pppssrspss spgqlgqppg elslglatgp lstpetpprq peppslvaga sggaspvgft 1021 prgglsppgh spgpprtfps apprasgshg slllppassp pppqvpqrrg tppltpgrlt 1081 qdlklisasq palpqdgaqt lrrasphssg esmaafplfp ragggsggsg ssgglgppgr 1141 pygaipgqhv tlprktssgs lppplslfga ratssggppl tagpqrepga rpepvrsklp 1201 snl';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cardiomyocyte\_2();

cells\_list.push(['cardiomyocyte 2', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+30, spawnery - 1.23758546e+30, spawnerx - 1.23758546e+30]);

}

function intercalated\_disks() {

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.187927353e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.856378206e+29) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = ' 1 mgdwsalgkl ldkvqaysta ggkvwlsvlf ifrilllgta vesawgdeqs afrcntqqpg 61 cenvcydksf pishvrfwvl qiifvsvptl lylahvfyvm rkeeklnkke eelkvaqtdg 121 vnvdmhlkqi eikkfkygie ehgkvkmrgg llrtyiisil fksifevafl liqwyiygfs 181 lsavytckrd pcphqvdcfl srptektifi ifmlvvslvs lalniielfy vffkgvkdrv 241 kgksdpyhat sgalspakdc gsqkyayfng cssptaplsp msppgyklvt gdrnnsscrn 301 ynkqaseqnw anysaeqnrm gqagstisns haqpfdfpdd nqnskklaag helqplaivd 361 qrpssrassr assrprpddl ei';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mgdwgflekl ldqvqehstv vgkiwltvlf ifrililgla gesvwgdeqs dfecntaqpg 61 ctnvcydqaf pishirywvl qflfvstptl vylghviyls rreerlrqke gelralpakd 121 pqveralaav erqmakisva edgrlrirga lmgtyvasvl cksvleagfl ygqwrlygwt 181 mepvfvcqra pcpylvdcfv srptektifi ifmlvvglis lvlnllelvh llcrclsrgm 241 rarqgqdapp tqgtssdpyt dqvffylpvg qgpssppcpt ynglssseqn wanltteerl 301 assrpplfld pppqngqkpp srpsssaskk qyv';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mgdwsflgnf leevhkhstv vgkvwltvlf ifrmlvlgta aesswgdeqa dfrcdtiqpg 61 cqnvcydqaf pishirywvl qiifvstpsl vymghamhtv rmqekrklre aerakevrgs 121 gsyeypvaek aelscweegn grialqgtll ntyvcsilir ttmevgfivg qyfiygiflt 181 tlhvcrrspc phpvncyvsr pteknvfivf mlavaalsll lslaelyhlg wkkirqrfvk 241 prqhmakcql sgpsvgivqs ctpppdfnqc lengpggkff npfsnnmasq qntdnlvteq 301 vrgqeqtpge gfiqvrygqk pevpngvspg hrlphgyhsd krrlskassk arsddlsv';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mswsfltrll eeihnhstfv gkiwltvliv frivltavgg esiyydeqsk fvcnteqpgc 61 envcydafap lshvrfwvfq iilvatpsvm ylgyaihkia kmehgeadkk aarskpyamr 121 wkqhraleet eedneedpmm ypemelesdk enkeqsqpkp khdgrrrire dglmkiyvlq 181 llartvfevg fligqyflyg fqvhpfyvcs rlpcphkidc fisrptekti fllimygvtg 241 lclllniwem lhlgfgtird slnskrrele dpgaynypft wntpsappgy niavkpdqiq 301 ytelsnakia ykqnkantaq eqqygsheen lpadlealqr eirmaqerld lavqayshqn 361 nphgprekka kvgskagsnk stassksgdg ktsvwi';

protein();

} else {

peptidesequence = ' 1 mnwtglytll sgvnrhstai grvwlsvifi frimvlvvaa esvwgdekss ficntlqpgc 61 nsvcydqffp ishvrlwslq lilvstpall vamhvahqqh iekkmlrleg hgdplhleev 121 krhkvhisgt lwwtyvisvv frllfeavfm yvfyllypgy amvrlvkcdv ypcpntvdcf 181 vsrptektvf tvfmlaasgi ciilnvaevv yliiracarr aqrrsnppsr kgsgfghrls 241 peykqneink llseqdgslk dilrrspgtg aglaeksdrc sac';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 6.002289533e+30;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.187927353e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.794498932e+29) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = ' 1 mgdwsalgkl ldkvqaysta ggkvwlsvlf ifrilllgta vesawgdeqs afrcntqqpg 61 cenvcydksf pishvrfwvl qiifvsvptl lylahvfyvm rkeeklnkke eelkvaqtdg 121 vnvdmhlkqi eikkfkygie ehgkvkmrgg llrtyiisil fksifevafl liqwyiygfs 181 lsavytckrd pcphqvdcfl srptektifi ifmlvvslvs lalniielfy vffkgvkdrv 241 kgksdpyhat sgalspakdc gsqkyayfng cssptaplsp msppgyklvt gdrnnsscrn 301 ynkqaseqnw anysaeqnrm gqagstisns haqpfdfpdd nqnskklaag helqplaivd 361 qrpssrassr assrprpddl ei';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mgdwgflekl ldqvqehstv vgkiwltvlf ifrililgla gesvwgdeqs dfecntaqpg 61 ctnvcydqaf pishirywvl qflfvstptl vylghviyls rreerlrqke gelralpakd 121 pqveralaav erqmakisva edgrlrirga lmgtyvasvl cksvleagfl ygqwrlygwt 181 mepvfvcqra pcpylvdcfv srptektifi ifmlvvglis lvlnllelvh llcrclsrgm 241 rarqgqdapp tqgtssdpyt dqvffylpvg qgpssppcpt ynglssseqn wanltteerl 301 assrpplfld pppqngqkpp srpsssaskk qyv';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mgdwsflgnf leevhkhstv vgkvwltvlf ifrmlvlgta aesswgdeqa dfrcdtiqpg 61 cqnvcydqaf pishirywvl qiifvstpsl vymghamhtv rmqekrklre aerakevrgs 121 gsyeypvaek aelscweegn grialqgtll ntyvcsilir ttmevgfivg qyfiygiflt 181 tlhvcrrspc phpvncyvsr pteknvfivf mlavaalsll lslaelyhlg wkkirqrfvk 241 prqhmakcql sgpsvgivqs ctpppdfnqc lengpggkff npfsnnmasq qntdnlvteq 301 vrgqeqtpge gfiqvrygqk pevpngvspg hrlphgyhsd krrlskassk arsddlsv';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mswsfltrll eeihnhstfv gkiwltvliv frivltavgg esiyydeqsk fvcnteqpgc 61 envcydafap lshvrfwvfq iilvatpsvm ylgyaihkia kmehgeadkk aarskpyamr 121 wkqhraleet eedneedpmm ypemelesdk enkeqsqpkp khdgrrrire dglmkiyvlq 181 llartvfevg fligqyflyg fqvhpfyvcs rlpcphkidc fisrptekti fllimygvtg 241 lclllniwem lhlgfgtird slnskrrele dpgaynypft wntpsappgy niavkpdqiq 301 ytelsnakia ykqnkantaq eqqygsheen lpadlealqr eirmaqerld lavqayshqn 361 nphgprekka kvgskagsnk stassksgdg ktsvwi';

protein();

} else {

peptidesequence = ' 1 mnwtglytll sgvnrhstai grvwlsvifi frimvlvvaa esvwgdekss ficntlqpgc 61 nsvcydqffp ishvrlwslq lilvstpall vamhvahqqh iekkmlrleg hgdplhleev 121 krhkvhisgt lwwtyvisvv frllfeavfm yvfyllypgy amvrlvkcdv ypcpntvdcf 181 vsrptektvf tvfmlaasgi ciilnvaevv yliiracarr aqrrsnppsr kgsgfghrls 241 peykqneink llseqdgslk dilrrspgtg aglaeksdrc sac';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.435444447e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.187927353e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 meqkityriw rdtanwlein pdtgaistra eldredfehv knstytalii atdngspvat 61 gtgtlllils dvndnapipe prtiffcern pkpqviniid adlppntspf taelthgasa 121 nwtiqyndpt qesiilkpkm alevgdykin lklmdnqnkd qvttlevsvc dcegaagvcr 181 kaqpveaglq ipailgilgg ilallilill lllflrrrav vkepllpped dtrdnvyyyd 241 eegggeedqd fdlsqlhrgl darpevtrnd vaptlmsvpr ylprpanpde ignfidenlk 301 aadtdptapp ydsllvfdye gsgseaasls slnssesdkd qdydylnewg nrfkkladmy 361 gggedd';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 meqkityriw rdtanwlein pdtgaistra eldredfehv knstytalii atdngspvat 61 gtgtlllils dvndnapipe prtiffcern pkpqviniid adlppntspf taelthgasa 121 nwtiqyndpt qesiilkpkm alevgdykin lklmdnqnkd qvttlevsvc dcegaagvcr 181 kaqpveaglq ipailgilgg ilallilill lllflrrrav vkepllpped dtrdnvyyyd 241 eegggeedqd fdlsqlhrgl darpevtrnd vaptlmsvpr ylprpanpde ignfidenlk 301 aadtdptapp ydsllvfdye gsgseaasls slnssesdkd qdydylnewg nrfkkladmy 361 gggedd';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgpwsrslsa lllllqvssw lcqepepchp gfdaesytft vprrhlergr vlgrvnfedc 61 tgrqrtayfs ldtrfkvgtd gvitvkrplr fhnpqihflv yawdstyrkf stkvtlntvg 121 hhhrppphqa svsgiqaell tfpnsspglr rqkrdwvipp iscpenekgp fpknlvqiks 181 nkdkegkvfy sitgqgadtp pvgvfiiere tgwlkvtepl dreriatytl fshavssngn 241 avedpmeili tvtdqndnkp eftqevfkgs vmegalpgts vmevtatdad ddvntynaai 301 aytilsqdpe lpdknmftin rntgvisvvt tgldresfpt ytlvvqaadl qgeglsttat 361 avitvtdtnd nppifnptty kgqvpenean vvittlkvtd adapntpawe avytilnddg 421 gqfvvttnpv nndgilktak gldfeakqqy ilhvavtnvv pfevslttst atvtvdvldv 481 neapifvppe krvevsedfg vgqeitsyta qepdtfmeqk ityriwrdta nwleinpdtg 541 aistraeldr edfehvknst ytaliiatdn gspvatgtgt lllilsdvnd napipeprti 601 ffcernpkpq viniidadlp pntspftael thgasanwti qyndptqesi ilkpkmalev 661 gdykinlklm dnqnkdqvtt levsvcdceg aagvcrkaqp veaglqipai lgilggilal 721 lililllllf lrrravvkep llppeddtrd nvyyydeegg geedqdfdls qlhrgldarp 781 evtrndvapt lmsvprylpr panpdeignf idenlkaadt dptappydsl lvfdyegsgs 841 eaaslsslns sesdkdqdyd ylnewgnrfk kladmyggge dd';

protein();

} else {

peptidesequence = ' 1 mgpwsrslsa lllllqvssw lcqepepchp gfdaesytft vprrhlergr vlgrvnfedc 61 tgrqrtayfs ldtrfkvgtd gvitvkrplr fhnpqihflv yawdstyrkf stkvtlntvg 121 hhhrppphqa svsgiqaell tfpnsspglr rqkrdwvipp iscpenekgp fpknlvqiks 181 nkdkegkvfy sitgqgadtp pvgvfiiere tgwlkvtepl dreriatytl fshavssngn 241 avedpmeili tvtdqndnkp eftqevfkgs vmegalpgts vmevtatdad ddvntynaai 301 aytilsqdpe lpdknmftin rntgvisvvt tgldresfpt ytlvvqaadl qgeglsttat 361 avitvtdtnd nppifnptty kgqvpenean vvittlkvtd adapntpawe avytilnddg 421 gqfvvttnpv nndgilktak gldfeakqqy ilhvavtnvv pfevslttst atvtvdvldv 481 neapifvppe krvevsedfg vgqeitsyta qepdtfmeqk ityriwrdta nwleinpdtg 541 aistraeldr edfehvknst ytaliiatdn gspvatgtgt lllilsdvnd napipeprti 601 ffcernpkpq viniidadlp pntspftael thgasanwti qyndptqesi ilkpkmalev 661 gdykinlklm dnqnkdqvtt levsvcdceg aagvcrkaqp veaglqipai lgilggilal 721 lililllllf lrrravvkep llppeddtrd nvyyydeegg geedqdfdls qlhrgldarp 781 evtrndvapt lmsvprylpr panpdeignf idenlkaadt dptappydsl lvfdyegsgs 841 eaaslsslns sesdkdqdyd ylnewgnrfk kladmyggge dd';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mddsevesta silasvkeqe aqfekltral eeerrhvsaq lervrvspqd anplmangtl 61 trrhqngrfv gdadlerqkf sdlklngpqd hshllystip rmqepgqive tyteedpega 121 msvvsvetsd dgttrrtett vkkvvktvtt rtvqpvamgp dglpvdassv snnyiqtlgr 181 dfrkngnggp gpyvgqagta tlprnfhypp dgysrhyedg ypggsdnygs lsrvtrieer 241 yrpsmegyra psrqdvygpq pqvrvggssv dlhrfhpepy gleddqrsmg yddldygmms 301 dygtarrtgt psdprrrlrs yedmigeevp sdqyywapla qhergslasl dslrkggppp 361 pnwrqpelpe viamlgfrld avksnaaayl qhlcyrndkv ktdvrklkgi pvlvglldhp 421 kkevhlgacg alknisfgrd qdnkiaiknc dgvpalvrll rkardmdlte vitgtlwnls 481 shdsikmeiv dhalhaltde viiphsgwer epnedckprh iewesvltnt agclrnvsse 541 rsearrklre cdglvdalif ivqaeigqkd sdsklvencv cllrnlsyqv hreipqaery 601 qeaapnvann tgphaascfg akkgkgkkpi edpandtvdf pkrtspargy ellfqpevvr 661 iyisllkesk tpaileasag aiqnlcagrw tygryirsal rqekalsaia dlltneherv 721 vkaasgalrn lavdarnkel igkhaipnlv knlpggqqns swnfsedtvi silntinevi 781 aenleaakkl retqgieklv linksgnrse kevraaalvl qtiwgykelr kplekegwkk 841 sdfqvnlnna srsqsshsyd dstlplidrn qksdkkpdre eiqmsnmgsn tksldnnyst 901 pnergdhnrt ldrsgdlgdm eplkgttplm qki';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mevmnlmeqp ikvtewqqty tydsgihsga ntcvpsvssk gimeedeacg rqytlkkttt 61 ytqgvppsqg dleyqmstta rakrvreamc pgvsgedssl llatqvegqa tnlqrlaeps 121 qllksaivhl inyqddaela tralpeltkl lndedpvvvt kaamivnqls kkeasrralm 181 gspqlvaavv rtmqntsdld tarcttsilh nlshhregll aifksggipa lvrmlsspve 241 svlfyaittl hnlllyqega kmavrladgl qkmvpllnkn npkflaittd clqllaygnq 301 eskliilang gpqalvqimr nysyekllwt tsrvlkvlsv cpsnkpaive aggmqalgkh 361 ltsnsprlvq nclwtlrnls dvatkqegle svlkilvnql svddvnvltc atgtlsnltc 421 nnsknktlvt qnsgvealih ailragdkdd itepavcalr hltsrhpeae maqnsvrlny 481 gipaivklln qpnqwplvka tiglirnlal cpanhaplqe aaviprlvql lvkahqdaqr 541 hvaagtqqpy tdgvrmeeiv egctgalhil ardpmnrmei frlntiplfv qllyssveni 601 qrvaagvlce laqdkeaada idaegasapl mellhsrneg tatyaaavlf risedknpdy 661 rkrvsveltn slfkhdpaaw eaaqsmipin epygddmdat yrpmyssdvp ldplemhmdm 721 dgdypidtys dglrppypta dhmla';

protein();

} else {

peptidesequence = ' 1 mtavhagnin fkwdpkslei rtlaverlle plvtqvttlv ntnskgpsnk krgrskkahv 61 laasveqate nflekgdkia kesqflkeel vaavedvrkq gdlmkaaage faddpcssvk 121 rgnmvraara llsavtrlli ladmadvykl lvqlkvvedg ilklrnagne qdlgiqykal 181 kpevdklnim aakrqqelkd vghrdqmaaa rgilqknvpi lytasqaclq hpdvaaykan 241 rdliykqlqq avtgisnaaq atasddasqh qgggggelay alnnfdkqii vdplsfseer 301 frpsleerle siisgaalma dssctrddrr erivaecnav rqalqdllse ymgnagrker 361 sdalnsaidk mtkktrdlrr qlrkavmdhv sdsfletnvp llvlieaakn gnekevkeya 421 qvfrehankl ievanlacsi snneegvklv rmsasqleal cpqvinaala laakpqskla 481 qenmdlfkeq wekqvrvltd avdditsidd flavsenhil edvnkcvial qekdvdgldr 541 tagairgraa rvihvvtsem dnyepgvyte kvleatklls ntvmprfteq veaavealss 601 dpaqpmdene fidasrlvyd girdirkavl mirtpeeldd sdfetedfdv rsrtsvqted 661 dqliagqsar aimaqlpqeq kakiaeqvas fqeeksklda evskwddsgn diivlakqmc 721 mimmemtdft rgkgplknts dvisaakkia eagsrmdklg rtiadhcpds ackqdllayl 781 qrialychql nicskvkaev qnlggelvvs gvdsamsliq aaknlmnavv qtvkasyvas 841 tkyqksqgma slnlpavswk mkapekkplv krekqdetqt kikrasqkkh vnpvqalsef 901 kamdsi';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 6.12604808e+30;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.435444447e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.569134618e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 meqkityriw rdtanwlein pdtgaistra eldredfehv knstytalii atdngspvat 61 gtgtlllils dvndnapipe prtiffcern pkpqviniid adlppntspf taelthgasa 121 nwtiqyndpt qesiilkpkm alevgdykin lklmdnqnkd qvttlevsvc dcegaagvcr 181 kaqpveaglq ipailgilgg ilallilill lllflrrrav vkepllpped dtrdnvyyyd 241 eegggeedqd fdlsqlhrgl darpevtrnd vaptlmsvpr ylprpanpde ignfidenlk 301 aadtdptapp ydsllvfdye gsgseaasls slnssesdkd qdydylnewg nrfkkladmy 361 gggedd';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 meqkityriw rdtanwlein pdtgaistra eldredfehv knstytalii atdngspvat 61 gtgtlllils dvndnapipe prtiffcern pkpqviniid adlppntspf taelthgasa 121 nwtiqyndpt qesiilkpkm alevgdykin lklmdnqnkd qvttlevsvc dcegaagvcr 181 kaqpveaglq ipailgilgg ilallilill lllflrrrav vkepllpped dtrdnvyyyd 241 eegggeedqd fdlsqlhrgl darpevtrnd vaptlmsvpr ylprpanpde ignfidenlk 301 aadtdptapp ydsllvfdye gsgseaasls slnssesdkd qdydylnewg nrfkkladmy 361 gggedd';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgpwsrslsa lllllqvssw lcqepepchp gfdaesytft vprrhlergr vlgrvnfedc 61 tgrqrtayfs ldtrfkvgtd gvitvkrplr fhnpqihflv yawdstyrkf stkvtlntvg 121 hhhrppphqa svsgiqaell tfpnsspglr rqkrdwvipp iscpenekgp fpknlvqiks 181 nkdkegkvfy sitgqgadtp pvgvfiiere tgwlkvtepl dreriatytl fshavssngn 241 avedpmeili tvtdqndnkp eftqevfkgs vmegalpgts vmevtatdad ddvntynaai 301 aytilsqdpe lpdknmftin rntgvisvvt tgldresfpt ytlvvqaadl qgeglsttat 361 avitvtdtnd nppifnptty kgqvpenean vvittlkvtd adapntpawe avytilnddg 421 gqfvvttnpv nndgilktak gldfeakqqy ilhvavtnvv pfevslttst atvtvdvldv 481 neapifvppe krvevsedfg vgqeitsyta qepdtfmeqk ityriwrdta nwleinpdtg 541 aistraeldr edfehvknst ytaliiatdn gspvatgtgt lllilsdvnd napipeprti 601 ffcernpkpq viniidadlp pntspftael thgasanwti qyndptqesi ilkpkmalev 661 gdykinlklm dnqnkdqvtt levsvcdceg aagvcrkaqp veaglqipai lgilggilal 721 lililllllf lrrravvkep llppeddtrd nvyyydeegg geedqdfdls qlhrgldarp 781 evtrndvapt lmsvprylpr panpdeignf idenlkaadt dptappydsl lvfdyegsgs 841 eaaslsslns sesdkdqdyd ylnewgnrfk kladmyggge dd';

protein();

} else {

peptidesequence = ' 1 mgpwsrslsa lllllqvssw lcqepepchp gfdaesytft vprrhlergr vlgrvnfedc 61 tgrqrtayfs ldtrfkvgtd gvitvkrplr fhnpqihflv yawdstyrkf stkvtlntvg 121 hhhrppphqa svsgiqaell tfpnsspglr rqkrdwvipp iscpenekgp fpknlvqiks 181 nkdkegkvfy sitgqgadtp pvgvfiiere tgwlkvtepl dreriatytl fshavssngn 241 avedpmeili tvtdqndnkp eftqevfkgs vmegalpgts vmevtatdad ddvntynaai 301 aytilsqdpe lpdknmftin rntgvisvvt tgldresfpt ytlvvqaadl qgeglsttat 361 avitvtdtnd nppifnptty kgqvpenean vvittlkvtd adapntpawe avytilnddg 421 gqfvvttnpv nndgilktak gldfeakqqy ilhvavtnvv pfevslttst atvtvdvldv 481 neapifvppe krvevsedfg vgqeitsyta qepdtfmeqk ityriwrdta nwleinpdtg 541 aistraeldr edfehvknst ytaliiatdn gspvatgtgt lllilsdvnd napipeprti 601 ffcernpkpq viniidadlp pntspftael thgasanwti qyndptqesi ilkpkmalev 661 gdykinlklm dnqnkdqvtt levsvcdceg aagvcrkaqp veaglqipai lgilggilal 721 lililllllf lrrravvkep llppeddtrd nvyyydeegg geedqdfdls qlhrgldarp 781 evtrndvapt lmsvprylpr panpdeignf idenlkaadt dptappydsl lvfdyegsgs 841 eaaslsslns sesdkdqdyd ylnewgnrfk kladmyggge dd';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mddsevesta silasvkeqe aqfekltral eeerrhvsaq lervrvspqd anplmangtl 61 trrhqngrfv gdadlerqkf sdlklngpqd hshllystip rmqepgqive tyteedpega 121 msvvsvetsd dgttrrtett vkkvvktvtt rtvqpvamgp dglpvdassv snnyiqtlgr 181 dfrkngnggp gpyvgqagta tlprnfhypp dgysrhyedg ypggsdnygs lsrvtrieer 241 yrpsmegyra psrqdvygpq pqvrvggssv dlhrfhpepy gleddqrsmg yddldygmms 301 dygtarrtgt psdprrrlrs yedmigeevp sdqyywapla qhergslasl dslrkggppp 361 pnwrqpelpe viamlgfrld avksnaaayl qhlcyrndkv ktdvrklkgi pvlvglldhp 421 kkevhlgacg alknisfgrd qdnkiaiknc dgvpalvrll rkardmdlte vitgtlwnls 481 shdsikmeiv dhalhaltde viiphsgwer epnedckprh iewesvltnt agclrnvsse 541 rsearrklre cdglvdalif ivqaeigqkd sdsklvencv cllrnlsyqv hreipqaery 601 qeaapnvann tgphaascfg akkgkgkkpi edpandtvdf pkrtspargy ellfqpevvr 661 iyisllkesk tpaileasag aiqnlcagrw tygryirsal rqekalsaia dlltneherv 721 vkaasgalrn lavdarnkel igkhaipnlv knlpggqqns swnfsedtvi silntinevi 781 aenleaakkl retqgieklv linksgnrse kevraaalvl qtiwgykelr kplekegwkk 841 sdfqvnlnna srsqsshsyd dstlplidrn qksdkkpdre eiqmsnmgsn tksldnnyst 901 pnergdhnrt ldrsgdlgdm eplkgttplm qki';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mevmnlmeqp ikvtewqqty tydsgihsga ntcvpsvssk gimeedeacg rqytlkkttt 61 ytqgvppsqg dleyqmstta rakrvreamc pgvsgedssl llatqvegqa tnlqrlaeps 121 qllksaivhl inyqddaela tralpeltkl lndedpvvvt kaamivnqls kkeasrralm 181 gspqlvaavv rtmqntsdld tarcttsilh nlshhregll aifksggipa lvrmlsspve 241 svlfyaittl hnlllyqega kmavrladgl qkmvpllnkn npkflaittd clqllaygnq 301 eskliilang gpqalvqimr nysyekllwt tsrvlkvlsv cpsnkpaive aggmqalgkh 361 ltsnsprlvq nclwtlrnls dvatkqegle svlkilvnql svddvnvltc atgtlsnltc 421 nnsknktlvt qnsgvealih ailragdkdd itepavcalr hltsrhpeae maqnsvrlny 481 gipaivklln qpnqwplvka tiglirnlal cpanhaplqe aaviprlvql lvkahqdaqr 541 hvaagtqqpy tdgvrmeeiv egctgalhil ardpmnrmei frlntiplfv qllyssveni 601 qrvaagvlce laqdkeaada idaegasapl mellhsrneg tatyaaavlf risedknpdy 661 rkrvsveltn slfkhdpaaw eaaqsmipin epygddmdat yrpmyssdvp ldplemhmdm 721 dgdypidtys dglrppypta dhmla';

protein();

} else {

peptidesequence = ' 1 mtavhagnin fkwdpkslei rtlaverlle plvtqvttlv ntnskgpsnk krgrskkahv 61 laasveqate nflekgdkia kesqflkeel vaavedvrkq gdlmkaaage faddpcssvk 121 rgnmvraara llsavtrlli ladmadvykl lvqlkvvedg ilklrnagne qdlgiqykal 181 kpevdklnim aakrqqelkd vghrdqmaaa rgilqknvpi lytasqaclq hpdvaaykan 241 rdliykqlqq avtgisnaaq atasddasqh qgggggelay alnnfdkqii vdplsfseer 301 frpsleerle siisgaalma dssctrddrr erivaecnav rqalqdllse ymgnagrker 361 sdalnsaidk mtkktrdlrr qlrkavmdhv sdsfletnvp llvlieaakn gnekevkeya 421 qvfrehankl ievanlacsi snneegvklv rmsasqleal cpqvinaala laakpqskla 481 qenmdlfkeq wekqvrvltd avdditsidd flavsenhil edvnkcvial qekdvdgldr 541 tagairgraa rvihvvtsem dnyepgvyte kvleatklls ntvmprfteq veaavealss 601 dpaqpmdene fidasrlvyd girdirkavl mirtpeeldd sdfetedfdv rsrtsvqted 661 dqliagqsar aimaqlpqeq kakiaeqvas fqeeksklda evskwddsgn diivlakqmc 721 mimmemtdft rgkgplknts dvisaakkia eagsrmdklg rtiadhcpds ackqdllayl 781 qrialychql nicskvkaev qnlggelvvs gvdsamsliq aaknlmnavv qtvkasyvas 841 tkyqksqgma slnlpavswk mkapekkplv krekqdetqt kikrasqkkh vnpvqalsef 901 kamdsi';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 6.187927353e+28;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.435444447e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.569134618e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 6.002289533e+30;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.435444447e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.569134618e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.682961542e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.187927353e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 75);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = ' 1 marspgraya lllllicfnv gsglhlqvls trnenkllpk hphlvrqkra witapvalre 61 gedlskknpi akihsdlaee rglkitykyt gkgiteppfg ifvfnkdtge lnvtsildre 121 etpfflltgy aldargnnve kplelrikvl dindnepvft qdvfvgsvee lsaahtlvmk 181 inatdadepn tlnskisyri vslepayppv fylnkdtgei yttsvtldre ehssytltve 241 ardgngevtd kpvkqaqvqi rildvndnip vvenkvlegm veenqvnvev trikvfdade 301 igsdnwlanf tfasgneggy fhietdaqtn egivtlikev dyeemknldf svivankaaf 361 hksirskykp tpipikvkvk nvkegihfks svisiyvses mdrsskgqii gnfqafdedt 421 glpaharyvk ledrdnwisv dsvtseikla klpdfesryv qngtytvkiv aisedyprkt 481 itgtvlinve dindncptli epvqtichda eyvnvtaedl dghpnsgpfs fsvidkppgm 541 aekwkiarqe stsvllqqse kklgrseiqf lisdnqgfsc pekqvltltv ceclhgsgcr 601 eaqhdsyvgl gpaaialmil aflllllvpl lllmchcgkg akgftpipgt iemlhpwnne 661 gappedkvvp sflpvdqggs lvgrngvggm akeatmkgss sasivkgqhe msemdgrwee 721 hrsllsgrat qftgatgaim ttettktara tgasrdmaga qaaavalnee flrnyftdka 781 asyteedenh takdcllvys qeeteslnas igccsfiege lddrflddlg lkfktlaevc 841 lgqkidinke ieqrqkpate tsmntashsl ceqtmvnsen tyssgssfpv pkslqeanae 901 kvtqeivter svssrqaqkv atplpdpmas rnviatetsy vtgstmpptt vilgpsqpqs 961 livtervyap astlvdqpya negtvvvter viqphgggsn plegtqhlqd vpyvmvrere 1021 sflapssgvq ptlampniav gqnvtvterv lapastlqss yqiptensmt arnttvsgag 1081 vpgplpdfgl eesghsnsti ttsstrvtkh stvqhsys';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 marspgraya lllllicfnv gsglhlqvls trnenkllpk hphlvrqkra witapvalre 61 gedlskknpi akihsdlaee rglkitykyt gkgiteppfg ifvfnkdtge lnvtsildre 121 etpfflltgy aldargnnve kplelrikvl dindnepvft qdvfvgsvee lsaahtlvmk 181 inatdadepn tlnskisyri vslepayppv fylnkdtgei yttsvtldre ehssytltve 241 ardgngevtd kpvkqaqvqi rildvndnip vvenkvlegm veenqvnvev trikvfdade 301 igsdnwlanf tfasgneggy fhietdaqtn egivtlikev dyeemknldf svivankaaf 361 hksirskykp tpipikvkvk nvkegihfks svisiyvses mdrsskgqii gnfqafdedt 421 glpaharyvk ledrdnwisv dsvtseikla klpdfesryv qngtytvkiv aisedyprkt 481 itgtvlinve dindncptli epvqtichda eyvnvtaedl dghpnsgpfs fsvidkppgm 541 aekwkiarqe stsvllqqse kklgrseiqf lisdnqgfsc pekqvltltv ceclhgsgcr 601 eaqhdsyvgl gpaaialmil aflllllvpl lllmchcgkg akgftpipgt iemlhpwnne 661 gappedkvvp sflpvdqggs lvgrngvggm akeatmkgss sasivkgqhe msemdgrwee 721 hrsllsgrat qftgatgaim ttettktara tgasrdmaga qaaavalnee flrnyftdka 781 asyteedenh takdcllvys qeeteslnas igccsfiege lddrflddlg lkfktlaevc 841 lgqkidinke ieqrqkpate tsmntashsl ceqtmvnsen tyssgssfpv pkslqeanae 901 kvtqeivter svssrqaqkv atplpdpmas rnviatetsy vtgstmpptt vilgpsqpqs 961 livtervyap astlvdqpya negtvvvter viqphgggsn plegtqhlqd vpyvmvrere 1021 sflapssgvq ptlampniav gqnvtvterv lapastlqss yqiptensmt arnttvsgag 1081 vpgplpdfgl eesghsnsti ttsstrvtkh stvqhsys';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 meaarpsgsw ngalcrllll tlailifasd acknvtlhvp skldaeklvg rvnlkecfta 61 anlihssdpd fqiledgsvy ttntillsse krsftillsn tenqekkkif vflehqtkvl 121 kkrhtkekvl rrakrrwapi pcsmlenslg pfplflqqvq sdtaqnytiy ysirgpgvdq 181 eprnlfyver dtgnlyctrp vdreqyesfe iiafattpdg ytpelplpli ikiedendny 241 pifteetytf tifencrvgt tvgqvcatdk depdtmhtrl kysiigqvpp sptlfsmhpt 301 tgvitttssq ldrelidkyq lkikvqdmdg qyfglqttst ciiniddvnd hlptftrtsy 361 vtsveentvd veilrvtved kdlvntanwr anytilkgne ngnfkivtda ktnegvlcvv 421 kplnyeekqq milqigvvne apfsreaspr samstatvtv nvedqdegpe cnppiqtvrm 481 kenaevgtts ngykaydpet rsssgirykk ltdptgwvti dentgsikvf rsldreaeti 541 kngiynitvl asdqggrtct gtlgiilqdv ndnspfipkk tviickptms saeivavdpd 601 epihgppfdf slesstsevq rmwrlkaind taarlsyqnd ppfgsyvvpi tvrdrlgmss 661 vtsldvtlcd citendcthr vdpriggggv qlgkwailai llgiallfci lftlvcgasg 721 tskqpkvipd dlaqqnlivs nteapgddkv ysangfttqt vgasaqgvcg tvgsgikngg 781 qetiemvkgg hqtsescrga ghhhtldscr gghtevdncr ytysewhsft qprlgekvyl 841 cnqdenhkha qdyvltynye grgsvagsvg ccserqeedg lefldnlepk frtlaeacmk 901 r';

protein();

} else {

peptidesequence = ' 1 meaarpsgsw ngalcrllll tlailifasd acknvtlhvp skldaeklvg rvnlkecfta 61 anlihssdpd fqiledgsvy ttntillsse krsftillsn tenqekkkif vflehqtkvl 121 kkrhtkekvl rrakrrwapi pcsmlenslg pfplflqqvq sdtaqnytiy ysirgpgvdq 181 eprnlfyver dtgnlyctrp vdreqyesfe iiafattpdg ytpelplpli ikiedendny 241 pifteetytf tifencrvgt tvgqvcatdk depdtmhtrl kysiigqvpp sptlfsmhpt 301 tgvitttssq ldrelidkyq lkikvqdmdg qyfglqttst ciiniddvnd hlptftrtsy 361 vtsveentvd veilrvtved kdlvntanwr anytilkgne ngnfkivtda ktnegvlcvv 421 kplnyeekqq milqigvvne apfsreaspr samstatvtv nvedqdegpe cnppiqtvrm 481 kenaevgtts ngykaydpet rsssgirykk ltdptgwvti dentgsikvf rsldreaeti 541 kngiynitvl asdqggrtct gtlgiilqdv ndnspfipkk tviickptms saeivavdpd 601 epihgppfdf slesstsevq rmwrlkaind taarlsyqnd ppfgsyvvpi tvrdrlgmss 661 vtsldvtlcd citendcthr vdpriggggv qlgkwailai llgiallfci lftlvcgasg 721 tskqpkvipd dlaqqnlivs nteapgddkv ysangfttqt vgasaqgvcg tvgsgikngg 781 qetiemvkgg hqtsescrga ghhhtldscr gghtevdncr ytysewhsft qprlgekvyl 841 cnqdenhkha qdyvltynye grgsvagsvg ccserqeedg lefldnlepk frtlaeacmk 901 r';

protein();

}

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = ' 1 mevmnlmeqp ikvtewqqty tydsgihsga ntcvpsvssk gimeedeacg rqytlkkttt 61 ytqgvppsqg dleyqmstta rakrvreamc pgvsgedssl llatqvegqa tnlqrlaeps 121 qllksaivhl inyqddaela tralpeltkl lndedpvvvt kaamivnqls kkeasrralm 181 gspqlvaavv rtmqntsdld tarcttsilh nlshhregll aifksggipa lvrmlsspve 241 svlfyaittl hnlllyqega kmavrladgl qkmvpllnkn npkflaittd clqllaygnq 301 eskliilang gpqalvqimr nysyekllwt tsrvlkvlsv cpsnkpaive aggmqalgkh 361 ltsnsprlvq nclwtlrnls dvatkqegle svlkilvnql svddvnvltc atgtlsnltc 421 nnsknktlvt qnsgvealih ailragdkdd itepavcalr hltsrhpeae maqnsvrlny 481 gipaivklln qpnqwplvka tiglirnlal cpanhaplqe aaviprlvql lvkahqdaqr 541 hvaagtqqpy tdgvrmeeiv egctgalhil ardpmnrmei frlntiplfv qllyssveni 601 qrvaagvlce laqdkeaada idaegasapl mellhsrneg tatyaaavlf risedknpdy 661 rkrvsveltn slfkhdpaaw eaaqsmipin epygddmdat yrpmyssdvp ldplemhmdm 721 dgdypidtys dglrppypta dhmla';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mscnggshpr intlgrmira esgpdlryev tsggggtsrm yysrrgvitd qnsdgycqtg 61 tmsrhqnqnt iqellqncsd clmraelivq pelkygdgiq ltrsreldec faqandqmei 121 ldsliremrq mgqpcdayqk rllqlqeqmr alykaisvpr vrrasskggg gytcqsgsgw 181 deftkhvtse clgwmrqqra emdmvawgvd lasveqhins hrgihnsigd yrwqldkika 241 dlreksaiyq leeeyenllk asfermdhlr qlqniiqats reimwindce eeellydwsd 301 kntniaqkqe afsirmsqle vkekelnklk qesdqlvlnq hpasdkieay mdtlqtqwsw 361 ilqitkcidv hlkenaayfq ffeeaqstea ylkglqdsir kkypcdknmp lqhlleqike 421 lekerekile ykrqvqnlvn kskkivqlkp rnpdyrsnkp iilralcdyk qdqkivhkgd 481 ecilkdnner skwyvtgpgg vdmlvpsvgl iipppnplav dlsckieqyy eailalwnql 541 yinmkslvsw hycmidieki ramtiaklkt mrqedymkti adlelhyqef irnsqgsemf 601 gdddkrkiqs qftdaqkhyq tlviqlpgyp qhqtvtttei thhgtcqdvn hnkvietnre 661 ndkqetwmlm elqkirrqie hcegrmtlkn lpladqgssh hitvkinelk svqndsqaia 721 evlnqlkdml anfrgsekyc ylqnevfglf qkleningvt dgylnslctv rallqailqt 781 edmlkvyear lteeetvcld ldkveayrcg lkkikndlnl kksllatmkt elqkaqqihs 841 qtsqqyplyd ldlgkfgekv tqltdrwqri dkqidfrlwd lekqikqlrn yrdnyqafck 901 wlydakrrqd slesmkfgds ntvmrflneq knlhseisgk rdkseevqki aelcansikd 961 yelqlasyts gletllnipi krtmiqspsg vilqeaadvh aryielltrs gdyyrflsem 1021 lksledlklk ntkievleee lrlardanse ncnknkfldq nlqkyqaecs qfkaklasle 1081 elkrqaeldg ksakqnldkc ygqikelnek itrltyeied ekrrrksved rfdqqkndyd 1141 qlqkarqcek enlgwqkles ekaikekeye ierlrvllqe egtrkreyen elakvrnhyn 1201 eemsnlrnky eteinitktt ikeismqked dsknlrnqld rlsrenrdlk deivrlndsi 1261 lqateqrrra eenalqqkac gseimqkkqh leielkqvmq qrsednarhk qsleeaakti 1321 qdknkeierl kaefqeeakr rweyenelsk vrnnydeeii slknqfetei nitkttihql 1381 tmqkeedtsg yraqidnltr enrslseeik rlkntltqtt enlrrveedi qqqkatgsev 1441 sqrkqqleve lrqvtqmrte esvrykqsld daaktiqdkn keierlkqli dketndrkcl 1501 edenarlqrv qydlqkanss atetinklkv qeqeltrlri dyervsqert vkdqditrfq 1561 nslkelqlqk qkveeelnrl krtasedsck rkkleeeleg mrrslkeqai kitnltqqle 1621 qasivkkrse ddlrqqrdvl dghlrekqrt qeelrrlsse vealrrqllq eqesvkqahl 1681 rnehfqkaie dksrslnesk ieierlqslt enltkehlml eeelrnlrle yddlrrgrse 1741 adsdknatil elrsqlqisn nrtlelqgli ndlqrerenl rqeiekfqkq aleasnriqe 1801 sknqctqvvq eresllvkik vleqdkarlq rledelnrak stleaetrvk qrlecekqqi 1861 qndlnqwktq ysrkeeairk ieserekser eknslrseie rlqaeikrie ercrrkleds 1921 tretqsqlet ersryqreid klrqrpygsh retqtecewt vdtsklvfdg lrkkvtamql 1981 yecqlidktt ldkllkgkks veevaseiqp flrgagsiag asaspkekys lveakrkkli 2041 spestvmlle aqaatggiid phrnekltvd saiardlidf ddrqqiyaae kaitgfddpf 2101 sgktvsvsea ikknlidret gmrlleaqia sggvvdpvns vflpkdvala rglidrdlyr 2161 slndprdsqk nfvdpvtkkk vsyvqlkerc riephtglll lsvqkrsmsf qgirqpvtvt 2221 elvdsgilrp stvnelesgq isydevgeri kdflqgssci agiynettkq klgiyeamki 2281 glvrpgtale lleaqaatgf ivdpvsnlrl pveeaykrgl vgiefkekll saeravtgyn 2341 dpetgniisl fqamnkelie kghgirllea qiatggiidp keshrlpvdi aykrgyfnee 2401 lseilsdpsd dtkgffdpnt eenltylqlk ercikdeetg lcllplkekk kqvqtsqknt 2461 lrkrrvvivd petnkemsvq eaykkglidy etfkelceqe ceweeititg sdgstrvvlv 2521 drktgsqydi qdaidkglvd rkffdqyrsg slsltqfadm islkngvgts ssmgsgvsdd 2581 vfsssrhesv skistissvr nltirsssfs dtleesspia aifdtenlek isitegierg 2641 ivdsitgqrl leaqactggi ihpttgqkls lqdavsqgvi dqdmatrlkp aqkafigfeg 2701 vkgkkkmsaa eavkekwlpy eagqrflefq yltgglvdpe vhgristeea irkgfidgra 2761 aqrlqdtssy akiltcpktk lkisykdain rsmveditgl rlleaasvss kglpspynms 2821 sapgsrsgsr sgsrsgsrsg srsgsrrgsf datgnssysy sysfssssig h';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 maapgapaey gyirtvlgqq ilgqldsssl alpseaklkl agssgrggqt vkslriqeqv 61 qqtlarkgrs svgngnlhrt ssvpeyvynl hlvendfvgg rspvpktydm lkagttatye 121 grwgrgtaqy ssqksveers lrhplrrlei spdssperah ythsdyqysq rsqaghtlhh 181 qesrraallv ppryarseiv gvsragttsr qrhfdtyhrq yqhgsvsdtv fdsipanpal 241 ltyprpgtsr smgnlleken yltagltvgq vrplvplqpv tqnrasrssw hqssfhstrt 301 lreagpsvav dssgrrahlt vgqaaaggsg nllterstft dsqlgnadme mtleravsml 361 eadhmlpsri saaatfiqhe cfqksearkr vnqlrgilkl lqllkvqned vqravcgalr 421 nlvfedndnk levaelngvp rllqvlkqtr dletkkqitg llwnlssndk lknlmiteal 481 ltlteniiip fsgwpegdyp kanglldfdi fynvtgclrn mssagadgrk amrrcdglid 541 slvhyvrgti adyqpddkat encvcilhnl syqleaelpe kysqniyiqn rniqtdnnks 601 igcfgsrsrk vkeqyqdvpm peeksnpkgv ewlwhsivir mylsliaksv rnytqeaslg 661 alqnltagsg pmptsvaqtv vqkesglqht rkmlhvgdps vkktaisllr nlsrnlslqn 721 eiaketlpdl vsiipdtvps tdlliettas acytlnniiq nsyqnardll ntggiqkima 781 isagdayasn kaskaasvll yslwahtelh haykkaqfkk tdfvnsrtak ayhslkd';

protein();

} else {

peptidesequence = ' 1 mddsevesta silasvkeqe aqfekltral eeerrhvsaq lervrvspqd anplmangtl 61 trrhqngrfv gdadlerqkf sdlklngpqd hshllystip rmqepgqive tyteedpega 121 msvvsvetsd dgttrrtett vkkvvktvtt rtvqpvamgp dglpvdassv snnyiqtlgr 181 dfrkngnggp gpyvgqagta tlprnfhypp dgysrhyedg ypggsdnygs lsrvtrieer 241 yrpsmegyra psrqdvygpq pqvrvggssv dlhrfhpepy gleddqrsmg yddldygmms 301 dygtarrtgt psdprrrlrs yedmigeevp sdqyywapla qhergslasl dslrkggppp 361 pnwrqpelpe viamlgfrld avksnaaayl qhlcyrndkv ktdvrklkgi pvlvglldhp 421 kkevhlgacg alknisfgrd qdnkiaiknc dgvpalvrll rkardmdlte vitgtlwnls 481 shdsikmeiv dhalhaltde viiphsgwer epnedckprh iewesvltnt agclrnvsse 541 rsearrklre cdglvdalif ivqaeigqkd sdsklvencv cllrnlsyqv hreipqaery 601 qeaapnvann tgphaascfg akkgkgkkpi edpandtvdf pkrtspargy ellfqpevvr 661 iyisllkesk tpaileasag aiqnlcagrw tygryirsal rqekalsaia dlltneherv 721 vkaasgalrn lavdarnkel igkhaipnlv knlpggqqns swnfsedtvi silntinevi 781 aenleaakkl retqgieklv linksgnrse kevraaalvl qtiwgykelr kplekegwkk 841 sdfqvnlnna srsqsshsyd dstlplidrn qksdkkpdre eiqmsnmgsn tksldnnyst 901 pnergdhnrt ldrsgdlgdm eplkgttplm qki';

protein();

}

} else if (randomizer <= 75) {

peptidesequence = '1 matqadlmel dmamepdrka avshwqqqsy ldsgihsgat ttapslsgkg npeeedvdts 61 qvlyeweqgf sqsftqeqva didgqyamtr aqrvraamfp etldegmqip stqfdaahpt 121 nvqrlaepsq mlkhavvnli nyqddaelat raipeltkll ndedqvvvnk aavmvhqlsk 181 keasrhaimr spqmvsaivr tmqntndvet arctagtlhn lshhreglla ifksggipal 241 vkmlgspvds vlfyaittlh nlllhqegak mavrlagglq kmvallnktn vkflaittdc 301 lqilaygnqe skliilasgg pqalvnimrt ytyekllwtt srvlkvlsvc ssnkpaivea 361 ggmqalglhl tdpsqrlvqn clwtlrnlsd aatkqegmeg llgtlvqllg sddinvvtca 421 agilsnltcn nyknkmmvcq vggiealvrt vlragdredi tepaicalrh ltsrhqeaem 481 aqnavrlhyg lpvvvkllhp pshwplikat vglirnlalc panhaplreq gaiprlvqll 541 vrahqdtqrr tsmggtqqqf vegvrmeeiv egctgalhil ardvhnrivi rglntiplfv 601 qllyspieni qrvaagvlce laqdkeaaea ieaegatapl tellhsrneg vatyaaavlf 661 rmsedkpqdy kkrlsvelts slfrtepmaw netadlgldi gaqgeplgyr qddpsyrsfh 721 sggygqdalg mdpmmehemg ghhpgadypv dglpdlghaq dlmdglppgd snqlawfdtd 781 l';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

cursor2z = spawnerz + 6.12604808e+30;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 6.682961542e+28;

repeat\_endcell1 = Math.round(149 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 2.475170941e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+28) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 75);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = ' 1 marspgraya lllllicfnv gsglhlqvls trnenkllpk hphlvrqkra witapvalre 61 gedlskknpi akihsdlaee rglkitykyt gkgiteppfg ifvfnkdtge lnvtsildre 121 etpfflltgy aldargnnve kplelrikvl dindnepvft qdvfvgsvee lsaahtlvmk 181 inatdadepn tlnskisyri vslepayppv fylnkdtgei yttsvtldre ehssytltve 241 ardgngevtd kpvkqaqvqi rildvndnip vvenkvlegm veenqvnvev trikvfdade 301 igsdnwlanf tfasgneggy fhietdaqtn egivtlikev dyeemknldf svivankaaf 361 hksirskykp tpipikvkvk nvkegihfks svisiyvses mdrsskgqii gnfqafdedt 421 glpaharyvk ledrdnwisv dsvtseikla klpdfesryv qngtytvkiv aisedyprkt 481 itgtvlinve dindncptli epvqtichda eyvnvtaedl dghpnsgpfs fsvidkppgm 541 aekwkiarqe stsvllqqse kklgrseiqf lisdnqgfsc pekqvltltv ceclhgsgcr 601 eaqhdsyvgl gpaaialmil aflllllvpl lllmchcgkg akgftpipgt iemlhpwnne 661 gappedkvvp sflpvdqggs lvgrngvggm akeatmkgss sasivkgqhe msemdgrwee 721 hrsllsgrat qftgatgaim ttettktara tgasrdmaga qaaavalnee flrnyftdka 781 asyteedenh takdcllvys qeeteslnas igccsfiege lddrflddlg lkfktlaevc 841 lgqkidinke ieqrqkpate tsmntashsl ceqtmvnsen tyssgssfpv pkslqeanae 901 kvtqeivter svssrqaqkv atplpdpmas rnviatetsy vtgstmpptt vilgpsqpqs 961 livtervyap astlvdqpya negtvvvter viqphgggsn plegtqhlqd vpyvmvrere 1021 sflapssgvq ptlampniav gqnvtvterv lapastlqss yqiptensmt arnttvsgag 1081 vpgplpdfgl eesghsnsti ttsstrvtkh stvqhsys';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 marspgraya lllllicfnv gsglhlqvls trnenkllpk hphlvrqkra witapvalre 61 gedlskknpi akihsdlaee rglkitykyt gkgiteppfg ifvfnkdtge lnvtsildre 121 etpfflltgy aldargnnve kplelrikvl dindnepvft qdvfvgsvee lsaahtlvmk 181 inatdadepn tlnskisyri vslepayppv fylnkdtgei yttsvtldre ehssytltve 241 ardgngevtd kpvkqaqvqi rildvndnip vvenkvlegm veenqvnvev trikvfdade 301 igsdnwlanf tfasgneggy fhietdaqtn egivtlikev dyeemknldf svivankaaf 361 hksirskykp tpipikvkvk nvkegihfks svisiyvses mdrsskgqii gnfqafdedt 421 glpaharyvk ledrdnwisv dsvtseikla klpdfesryv qngtytvkiv aisedyprkt 481 itgtvlinve dindncptli epvqtichda eyvnvtaedl dghpnsgpfs fsvidkppgm 541 aekwkiarqe stsvllqqse kklgrseiqf lisdnqgfsc pekqvltltv ceclhgsgcr 601 eaqhdsyvgl gpaaialmil aflllllvpl lllmchcgkg akgftpipgt iemlhpwnne 661 gappedkvvp sflpvdqggs lvgrngvggm akeatmkgss sasivkgqhe msemdgrwee 721 hrsllsgrat qftgatgaim ttettktara tgasrdmaga qaaavalnee flrnyftdka 781 asyteedenh takdcllvys qeeteslnas igccsfiege lddrflddlg lkfktlaevc 841 lgqkidinke ieqrqkpate tsmntashsl ceqtmvnsen tyssgssfpv pkslqeanae 901 kvtqeivter svssrqaqkv atplpdpmas rnviatetsy vtgstmpptt vilgpsqpqs 961 livtervyap astlvdqpya negtvvvter viqphgggsn plegtqhlqd vpyvmvrere 1021 sflapssgvq ptlampniav gqnvtvterv lapastlqss yqiptensmt arnttvsgag 1081 vpgplpdfgl eesghsnsti ttsstrvtkh stvqhsys';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 meaarpsgsw ngalcrllll tlailifasd acknvtlhvp skldaeklvg rvnlkecfta 61 anlihssdpd fqiledgsvy ttntillsse krsftillsn tenqekkkif vflehqtkvl 121 kkrhtkekvl rrakrrwapi pcsmlenslg pfplflqqvq sdtaqnytiy ysirgpgvdq 181 eprnlfyver dtgnlyctrp vdreqyesfe iiafattpdg ytpelplpli ikiedendny 241 pifteetytf tifencrvgt tvgqvcatdk depdtmhtrl kysiigqvpp sptlfsmhpt 301 tgvitttssq ldrelidkyq lkikvqdmdg qyfglqttst ciiniddvnd hlptftrtsy 361 vtsveentvd veilrvtved kdlvntanwr anytilkgne ngnfkivtda ktnegvlcvv 421 kplnyeekqq milqigvvne apfsreaspr samstatvtv nvedqdegpe cnppiqtvrm 481 kenaevgtts ngykaydpet rsssgirykk ltdptgwvti dentgsikvf rsldreaeti 541 kngiynitvl asdqggrtct gtlgiilqdv ndnspfipkk tviickptms saeivavdpd 601 epihgppfdf slesstsevq rmwrlkaind taarlsyqnd ppfgsyvvpi tvrdrlgmss 661 vtsldvtlcd citendcthr vdpriggggv qlgkwailai llgiallfci lftlvcgasg 721 tskqpkvipd dlaqqnlivs nteapgddkv ysangfttqt vgasaqgvcg tvgsgikngg 781 qetiemvkgg hqtsescrga ghhhtldscr gghtevdncr ytysewhsft qprlgekvyl 841 cnqdenhkha qdyvltynye grgsvagsvg ccserqeedg lefldnlepk frtlaeacmk 901 r';

protein();

} else {

peptidesequence = ' 1 meaarpsgsw ngalcrllll tlailifasd acknvtlhvp skldaeklvg rvnlkecfta 61 anlihssdpd fqiledgsvy ttntillsse krsftillsn tenqekkkif vflehqtkvl 121 kkrhtkekvl rrakrrwapi pcsmlenslg pfplflqqvq sdtaqnytiy ysirgpgvdq 181 eprnlfyver dtgnlyctrp vdreqyesfe iiafattpdg ytpelplpli ikiedendny 241 pifteetytf tifencrvgt tvgqvcatdk depdtmhtrl kysiigqvpp sptlfsmhpt 301 tgvitttssq ldrelidkyq lkikvqdmdg qyfglqttst ciiniddvnd hlptftrtsy 361 vtsveentvd veilrvtved kdlvntanwr anytilkgne ngnfkivtda ktnegvlcvv 421 kplnyeekqq milqigvvne apfsreaspr samstatvtv nvedqdegpe cnppiqtvrm 481 kenaevgtts ngykaydpet rsssgirykk ltdptgwvti dentgsikvf rsldreaeti 541 kngiynitvl asdqggrtct gtlgiilqdv ndnspfipkk tviickptms saeivavdpd 601 epihgppfdf slesstsevq rmwrlkaind taarlsyqnd ppfgsyvvpi tvrdrlgmss 661 vtsldvtlcd citendcthr vdpriggggv qlgkwailai llgiallfci lftlvcgasg 721 tskqpkvipd dlaqqnlivs nteapgddkv ysangfttqt vgasaqgvcg tvgsgikngg 781 qetiemvkgg hqtsescrga ghhhtldscr gghtevdncr ytysewhsft qprlgekvyl 841 cnqdenhkha qdyvltynye grgsvagsvg ccserqeedg lefldnlepk frtlaeacmk 901 r';

protein();

}

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = ' 1 mevmnlmeqp ikvtewqqty tydsgihsga ntcvpsvssk gimeedeacg rqytlkkttt 61 ytqgvppsqg dleyqmstta rakrvreamc pgvsgedssl llatqvegqa tnlqrlaeps 121 qllksaivhl inyqddaela tralpeltkl lndedpvvvt kaamivnqls kkeasrralm 181 gspqlvaavv rtmqntsdld tarcttsilh nlshhregll aifksggipa lvrmlsspve 241 svlfyaittl hnlllyqega kmavrladgl qkmvpllnkn npkflaittd clqllaygnq 301 eskliilang gpqalvqimr nysyekllwt tsrvlkvlsv cpsnkpaive aggmqalgkh 361 ltsnsprlvq nclwtlrnls dvatkqegle svlkilvnql svddvnvltc atgtlsnltc 421 nnsknktlvt qnsgvealih ailragdkdd itepavcalr hltsrhpeae maqnsvrlny 481 gipaivklln qpnqwplvka tiglirnlal cpanhaplqe aaviprlvql lvkahqdaqr 541 hvaagtqqpy tdgvrmeeiv egctgalhil ardpmnrmei frlntiplfv qllyssveni 601 qrvaagvlce laqdkeaada idaegasapl mellhsrneg tatyaaavlf risedknpdy 661 rkrvsveltn slfkhdpaaw eaaqsmipin epygddmdat yrpmyssdvp ldplemhmdm 721 dgdypidtys dglrppypta dhmla';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mscnggshpr intlgrmira esgpdlryev tsggggtsrm yysrrgvitd qnsdgycqtg 61 tmsrhqnqnt iqellqncsd clmraelivq pelkygdgiq ltrsreldec faqandqmei 121 ldsliremrq mgqpcdayqk rllqlqeqmr alykaisvpr vrrasskggg gytcqsgsgw 181 deftkhvtse clgwmrqqra emdmvawgvd lasveqhins hrgihnsigd yrwqldkika 241 dlreksaiyq leeeyenllk asfermdhlr qlqniiqats reimwindce eeellydwsd 301 kntniaqkqe afsirmsqle vkekelnklk qesdqlvlnq hpasdkieay mdtlqtqwsw 361 ilqitkcidv hlkenaayfq ffeeaqstea ylkglqdsir kkypcdknmp lqhlleqike 421 lekerekile ykrqvqnlvn kskkivqlkp rnpdyrsnkp iilralcdyk qdqkivhkgd 481 ecilkdnner skwyvtgpgg vdmlvpsvgl iipppnplav dlsckieqyy eailalwnql 541 yinmkslvsw hycmidieki ramtiaklkt mrqedymkti adlelhyqef irnsqgsemf 601 gdddkrkiqs qftdaqkhyq tlviqlpgyp qhqtvtttei thhgtcqdvn hnkvietnre 661 ndkqetwmlm elqkirrqie hcegrmtlkn lpladqgssh hitvkinelk svqndsqaia 721 evlnqlkdml anfrgsekyc ylqnevfglf qkleningvt dgylnslctv rallqailqt 781 edmlkvyear lteeetvcld ldkveayrcg lkkikndlnl kksllatmkt elqkaqqihs 841 qtsqqyplyd ldlgkfgekv tqltdrwqri dkqidfrlwd lekqikqlrn yrdnyqafck 901 wlydakrrqd slesmkfgds ntvmrflneq knlhseisgk rdkseevqki aelcansikd 961 yelqlasyts gletllnipi krtmiqspsg vilqeaadvh aryielltrs gdyyrflsem 1021 lksledlklk ntkievleee lrlardanse ncnknkfldq nlqkyqaecs qfkaklasle 1081 elkrqaeldg ksakqnldkc ygqikelnek itrltyeied ekrrrksved rfdqqkndyd 1141 qlqkarqcek enlgwqkles ekaikekeye ierlrvllqe egtrkreyen elakvrnhyn 1201 eemsnlrnky eteinitktt ikeismqked dsknlrnqld rlsrenrdlk deivrlndsi 1261 lqateqrrra eenalqqkac gseimqkkqh leielkqvmq qrsednarhk qsleeaakti 1321 qdknkeierl kaefqeeakr rweyenelsk vrnnydeeii slknqfetei nitkttihql 1381 tmqkeedtsg yraqidnltr enrslseeik rlkntltqtt enlrrveedi qqqkatgsev 1441 sqrkqqleve lrqvtqmrte esvrykqsld daaktiqdkn keierlkqli dketndrkcl 1501 edenarlqrv qydlqkanss atetinklkv qeqeltrlri dyervsqert vkdqditrfq 1561 nslkelqlqk qkveeelnrl krtasedsck rkkleeeleg mrrslkeqai kitnltqqle 1621 qasivkkrse ddlrqqrdvl dghlrekqrt qeelrrlsse vealrrqllq eqesvkqahl 1681 rnehfqkaie dksrslnesk ieierlqslt enltkehlml eeelrnlrle yddlrrgrse 1741 adsdknatil elrsqlqisn nrtlelqgli ndlqrerenl rqeiekfqkq aleasnriqe 1801 sknqctqvvq eresllvkik vleqdkarlq rledelnrak stleaetrvk qrlecekqqi 1861 qndlnqwktq ysrkeeairk ieserekser eknslrseie rlqaeikrie ercrrkleds 1921 tretqsqlet ersryqreid klrqrpygsh retqtecewt vdtsklvfdg lrkkvtamql 1981 yecqlidktt ldkllkgkks veevaseiqp flrgagsiag asaspkekys lveakrkkli 2041 spestvmlle aqaatggiid phrnekltvd saiardlidf ddrqqiyaae kaitgfddpf 2101 sgktvsvsea ikknlidret gmrlleaqia sggvvdpvns vflpkdvala rglidrdlyr 2161 slndprdsqk nfvdpvtkkk vsyvqlkerc riephtglll lsvqkrsmsf qgirqpvtvt 2221 elvdsgilrp stvnelesgq isydevgeri kdflqgssci agiynettkq klgiyeamki 2281 glvrpgtale lleaqaatgf ivdpvsnlrl pveeaykrgl vgiefkekll saeravtgyn 2341 dpetgniisl fqamnkelie kghgirllea qiatggiidp keshrlpvdi aykrgyfnee 2401 lseilsdpsd dtkgffdpnt eenltylqlk ercikdeetg lcllplkekk kqvqtsqknt 2461 lrkrrvvivd petnkemsvq eaykkglidy etfkelceqe ceweeititg sdgstrvvlv 2521 drktgsqydi qdaidkglvd rkffdqyrsg slsltqfadm islkngvgts ssmgsgvsdd 2581 vfsssrhesv skistissvr nltirsssfs dtleesspia aifdtenlek isitegierg 2641 ivdsitgqrl leaqactggi ihpttgqkls lqdavsqgvi dqdmatrlkp aqkafigfeg 2701 vkgkkkmsaa eavkekwlpy eagqrflefq yltgglvdpe vhgristeea irkgfidgra 2761 aqrlqdtssy akiltcpktk lkisykdain rsmveditgl rlleaasvss kglpspynms 2821 sapgsrsgsr sgsrsgsrsg srsgsrrgsf datgnssysy sysfssssig h';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 maapgapaey gyirtvlgqq ilgqldsssl alpseaklkl agssgrggqt vkslriqeqv 61 qqtlarkgrs svgngnlhrt ssvpeyvynl hlvendfvgg rspvpktydm lkagttatye 121 grwgrgtaqy ssqksveers lrhplrrlei spdssperah ythsdyqysq rsqaghtlhh 181 qesrraallv ppryarseiv gvsragttsr qrhfdtyhrq yqhgsvsdtv fdsipanpal 241 ltyprpgtsr smgnlleken yltagltvgq vrplvplqpv tqnrasrssw hqssfhstrt 301 lreagpsvav dssgrrahlt vgqaaaggsg nllterstft dsqlgnadme mtleravsml 361 eadhmlpsri saaatfiqhe cfqksearkr vnqlrgilkl lqllkvqned vqravcgalr 421 nlvfedndnk levaelngvp rllqvlkqtr dletkkqitg llwnlssndk lknlmiteal 481 ltlteniiip fsgwpegdyp kanglldfdi fynvtgclrn mssagadgrk amrrcdglid 541 slvhyvrgti adyqpddkat encvcilhnl syqleaelpe kysqniyiqn rniqtdnnks 601 igcfgsrsrk vkeqyqdvpm peeksnpkgv ewlwhsivir mylsliaksv rnytqeaslg 661 alqnltagsg pmptsvaqtv vqkesglqht rkmlhvgdps vkktaisllr nlsrnlslqn 721 eiaketlpdl vsiipdtvps tdlliettas acytlnniiq nsyqnardll ntggiqkima 781 isagdayasn kaskaasvll yslwahtelh haykkaqfkk tdfvnsrtak ayhslkd';

protein();

} else {

peptidesequence = ' 1 mddsevesta silasvkeqe aqfekltral eeerrhvsaq lervrvspqd anplmangtl 61 trrhqngrfv gdadlerqkf sdlklngpqd hshllystip rmqepgqive tyteedpega 121 msvvsvetsd dgttrrtett vkkvvktvtt rtvqpvamgp dglpvdassv snnyiqtlgr 181 dfrkngnggp gpyvgqagta tlprnfhypp dgysrhyedg ypggsdnygs lsrvtrieer 241 yrpsmegyra psrqdvygpq pqvrvggssv dlhrfhpepy gleddqrsmg yddldygmms 301 dygtarrtgt psdprrrlrs yedmigeevp sdqyywapla qhergslasl dslrkggppp 361 pnwrqpelpe viamlgfrld avksnaaayl qhlcyrndkv ktdvrklkgi pvlvglldhp 421 kkevhlgacg alknisfgrd qdnkiaiknc dgvpalvrll rkardmdlte vitgtlwnls 481 shdsikmeiv dhalhaltde viiphsgwer epnedckprh iewesvltnt agclrnvsse 541 rsearrklre cdglvdalif ivqaeigqkd sdsklvencv cllrnlsyqv hreipqaery 601 qeaapnvann tgphaascfg akkgkgkkpi edpandtvdf pkrtspargy ellfqpevvr 661 iyisllkesk tpaileasag aiqnlcagrw tygryirsal rqekalsaia dlltneherv 721 vkaasgalrn lavdarnkel igkhaipnlv knlpggqqns swnfsedtvi silntinevi 781 aenleaakkl retqgieklv linksgnrse kevraaalvl qtiwgykelr kplekegwkk 841 sdfqvnlnna srsqsshsyd dstlplidrn qksdkkpdre eiqmsnmgsn tksldnnyst 901 pnergdhnrt ldrsgdlgdm eplkgttplm qki';

protein();

}

} else if (randomizer <= 75) {

peptidesequence = '1 matqadlmel dmamepdrka avshwqqqsy ldsgihsgat ttapslsgkg npeeedvdts 61 qvlyeweqgf sqsftqeqva didgqyamtr aqrvraamfp etldegmqip stqfdaahpt 121 nvqrlaepsq mlkhavvnli nyqddaelat raipeltkll ndedqvvvnk aavmvhqlsk 181 keasrhaimr spqmvsaivr tmqntndvet arctagtlhn lshhreglla ifksggipal 241 vkmlgspvds vlfyaittlh nlllhqegak mavrlagglq kmvallnktn vkflaittdc 301 lqilaygnqe skliilasgg pqalvnimrt ytyekllwtt srvlkvlsvc ssnkpaivea 361 ggmqalglhl tdpsqrlvqn clwtlrnlsd aatkqegmeg llgtlvqllg sddinvvtca 421 agilsnltcn nyknkmmvcq vggiealvrt vlragdredi tepaicalrh ltsrhqeaem 481 aqnavrlhyg lpvvvkllhp pshwplikat vglirnlalc panhaplreq gaiprlvqll 541 vrahqdtqrr tsmggtqqqf vegvrmeeiv egctgalhil ardvhnrivi rglntiplfv 601 qllyspieni qrvaagvlce laqdkeaaea ieaegatapl tellhsrneg vatyaaavlf 661 rmsedkpqdy kkrlsvelts slfrtepmaw netadlgldi gaqgeplgyr qddpsyrsfh 721 sggygqdalg mdpmmehemg ghhpgadypv dglpdlghaq dlmdglppgd snqlawfdtd 781 l';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 7.425512824e+27;

}

}

function default\_cardiomyocyte\_1() {

if (unused\_variable == 5) {

cursor2z = spawnerz + 2.475170941e+26 \* 1;

cursor2z = spawnerz + 2.475170941e+26 \* 1;

cursor2x = spawnerx - 1 \* 6.187927353e+28;

lipid2 = 1.113826923e+30 - 0;

lipid1 = 1.113826923e+30 - 0;

lipid\_wall\_z();

cursor2z = spawnerz + 6.187679836e+30 \* 1;

lipid\_wall\_z();

lipid2 = 6.187927353e+30 - (4.950341882e+26 + 1);

lipid1 = 1.113826923e+30 - 0;

cursor2z = spawnerz + 2.475170941e+26 \* 1;

cursor2y = spawnery - 1 \* 6.187927353e+28;

cursor2x = spawnerx - 1 \* 6.187927353e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = spawnery - (1.175706197e+30 - 0);

lipid\_wall\_x();

cursor2y = spawnery - 1 \* 6.187927353e+26;

cursor1y = spawnery - 1 \* 6.249806627e+28;

cursor1x = spawnerx - 1 \* 6.249806627e+28;

cursor1z = spawnerz + 1.856378206e+29 \* 1;

for (countcell1 = 0; countcell1 < 7000; countcell1++) {

set\_earth\_rotation();

cursor1x = cursor1x - 1 \* 1.23758547e+29;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 mfpspaltpt pfsvkdilnl eqqqrslaaa gelsarleat lapsscmlaa fkpeayagpe 61 aaapglpelr aelgrapspa kcasafpaap afypraysdp dpakdpraek kelcalqkav 121 elekteadna erprarrrrk prvlfsqaqv yelerrfkqq rylsaperdq lasvlkltst 181 qvkiwfqnrr ykckrqrqdq tlelvglppp pppparriav pvlvrdgkpc lgdsapyapa 241 ygvglnpygy naypaypgyg gaacspgysc taaypagpsp aqpataaann nfvnfgvgdl 301 navqspgipq snsgvstlhg iraw';

protein();

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 myqslamaan hgpppgayea ggpgafmhga gaasspvyvp tprvpssvlg lsylqgggag 61 sasggasggs sggaasgagp gtqqgspgws qagadgaayt pppvsprfsf pgttgslaaa 121 aaaaaareaa ayssgggaag aglagreqyg ragfagsyss pypaymadvg aswaaaaaas 181 agpfdspvlh slpgranpaa rhpnlvdmfd dfsegrecvn cgamstplwr rdgtghylcn 241 acglyhkmng inrplikpqr rlsasrrvgl scancqtttt tlwrrnaege pvcnacglym 301 klhgvprpla mrkegiqtrk rkpknlnksk tpaapsgses lppasgassn ssnattssse 361 emrpiktepg lsshyghsss vsqtfsvsam sghgpsihpv lsalklspqg yaspvsqspq 421 tsskqdswns lvladshgdi ita';

} else if (randomizer <= 40) {

peptidesequence = '1 mgrkkiqitr imdernrqvt ftkrkfglmk kayelsvlcd ceialiifns tnklfqyast 61 dmdkvllkyt eynephesrt nsdivetlrk kglngcdspd pdaddsvghs pesedkyrki 121 nedidlmisr qrlcavpppn fempvsipvs shnslvysnp vsslgnpnll plahpslqrn 181 smspgvthrp psagntgglm ggdltsgagt sagngygnpr nspgllvspg nlnknmqaks 241 pppmnlgmnn rkpdlrvlip pgskntmpsv sedvdlllnq rinnsqsaqs latpvvsvat 301 ptlpgqgmgg ypsaisttyg teyslssadl sslsgfntas alhlgsvtgw qqqhlhnmpp 361 salsqlgact sthlsqssnl slpstqslni ksepvspprd rtttpsrypq htrheagrsp 421 vdslsscsss ydgsdredhr nefhspiglt rpspderesp svkrmrlseg wat';

} else if (randomizer <= 60) {

peptidesequence = '1 madadegfgl ahtplepdak dlpcdskpes algapsksps spqaaftqqg megikvflhe 61 relwlkfhev gtemiitkag rrmfpsykvk vtglnpktky illmdivpad dhrykfadnk 121 wsvtgkaepa mpgrlyvhpd spatgahwmr qlvsfqklkl tnnhldpfgh iilnsmhkyq 181 prlhivkade nngfgsknta fcthvfpeta fiavtsyqnh kitqlkienn pfakgfrgsd 241 dmelhrmsrm qskeypvvpr stvrqkvasn hspfssesra lstssnlgsq yqcengvsgp 301 sqdllpppnp yplpqehsqi yhctkrkeee csttdhpykk pymetspsee dsfyrssypq 361 qqglgasyrt esaqrqacmy assappsepv pslediscnt wpsmpsyssc tvttvqpmdr 421 lpyqhfsahf tsgplvprla gmanhgspql gegmfqhqts vahqpvvrqc gpqtglqspg 481 tlqppeflys hgvprtlsph qyhsvhgvgm vpewsdns';

} else if (randomizer <= 80) {

peptidesequence = '1 mdsvelclpe sfslhyeeel lcrmsnkdrh idsscssfik tepsspaslt dsvnhhspgg 61 ssdasgsyss tmnghqngld spplypsapi lggsgpvrkl yddcsstive dpqtkceyml 121 nsmpkrlclv cgdiasgyhy gvasceacka ffkrtiqgni eyscpatnec eitkrrrksc 181 qacrfmkclk vgmlkegvrl drvrggrqky krridaensp ylnpqlvqpa kkpynkivsh 241 llvaepekiy ampdptvpds dikalttlcd ladrelvvii gwakhipgfs tlsladqmsl 301 lqsawmeili lgvvyrslsf edelvyaddy imdedqskla glldlnnail qlvkkyksmk 361 lekeefvtlk aialansdsm hiedveavqk lqdvlhealq dyeagqhmed prragkmlmt 421 lpllrqtstk avqhfynikl egkvpmhklf lemleakv';

} else {

peptidesequence = ' 1 maqplcppls eswmlsaawg ptrrpppsdk dcgrslvssp dswgstpads pvasparpgt 61 lrdprapsvg rrgarssrlg sgqrqsaser eklrmrtlar alhelrrflp psvapagqsl 121 tkietlrlai ryighlsavl glseeslqrr crqrgdagsp rgcplcpddc paqmqtrtqa 181 egqgqgrglg lvsavragas wgsppacpga raapeprdpp alfaeaacpe gqamepspps 241 pllpgdvlal letwmplspl ewlpeepk';

}

protein();

}

}

cursor1x = spawnerx - 1 \* 5.940410259e+30;

cursor1y = spawnery - 1 \* 6.3116859e+28;

nucleoid\_size\_y = 5.878530986e+28;

nucleoid\_size\_z = 1.231397543e+30;

nucleoid();

cursor1z = spawnerz + 1 \* 1;

cursor1y = spawnery - 1 \* 1.23758547e+29;

cursor1x = spawnerx - 1 \* 1.23758547e+29;

vital\_components();

cursor1z = spawnerz + 1 \* 3.093963676e+29;

cursor1y = spawnery - 1 \* 1.23758547e+29;

cursor1x = spawnerx - 2 \* 1.23758547e+29;

vital\_components2();

cursor1z = spawnerz + 1 \* 3.093963676e+29;

cursor1y = spawnery - 1 \* 1.23758547e+29;

cursor1x = spawnerx - 2 \* 1.23758547e+29;

vital\_components3();

cursor1z = spawnerz + 6.187927353e+26 \* 1;

cursor1x = spawnerx - 1 \* 6.249806627e+28;

cursor1y = spawnery - (1.144147767e+30 - 0);

repeat\_endcell1 = Math.round(1.23758547e+30 / 6.497323721e+28 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(1.175706197e+30 / 3.712756412e+28 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

mitochondrion();

cursor1x = cursor1x - 3.712756412e+28;

}

cursor1x = cursor1x + 3.712756412e+28 \* Math.round(1.175706197e+30 / 3.712756412e+28 - 2);

cursor1z = cursor1z + 6.497323721e+28;

}

}

if (unused\_variable == 5) {

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(13 \* 1 - 0);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

if (unused\_variable == 5) {

cursor2x = cursor2x - 1.23758547e+28;

lipid2 = 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2x = cursor2x + 1.23758547e+28;

lipid1 = lipid2;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

lipid2 = 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid\_wall\_x();

cursor2x = cursor2x - 1.23758547e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.23758547e+28;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 93;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - 1.856378206e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 17;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - (2.351412394e+29 + (6.187927353e+28 \* 12 + 1.23758547e+28));

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(17 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 17;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(93 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(14 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 1.853903035e+29;

lipid1 = 1.113826923e+30;

lipid\_wall\_x();

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 6.187432319e+30;

lipid1 = 1.732619659e+29;

lipid\_wall\_x();

cursor2z = spawnerz + 2.475170941e+26;

cursor2y = spawnery - 1.05194765e+30;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 6.187432319e+30;

lipid1 = 1.23758547e+29;

lipid\_wall\_x();

cursor2z = spawnerz + 5.89090684e+30;

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 1.175706197e+30;

lipid2 = 2.967729958e+29;

lipid1 = 1.113826923e+30;

lipid\_wall\_x();

cursor2z = spawnerz + (1.856378206e+29 + 1.23758547e+28);

cursor2y = spawnery - 6.187927353e+28;

cursor2x = spawnerx - 1.175706197e+30;

repeat\_endcell1 = Math.round(93 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

lipid2 = 1.113826923e+30;

lipid1 = 1.236966678e+29 - 1.23758547e+28;

lipid\_wall\_x();

}

cursor2z = cursor2z + 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (2.475170941e+26 + 0);

cursor2y = spawnery - 2.475170941e+29;

cursor2x = spawnerx - 1.175706197e+30;

repeat\_endcell1 = Math.round(13 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

if (unused\_variable == 5) {

lipid2 = 1.113826923e+30;

lipid1 = 4.950341882e+28;

lipid\_wall\_x();

}

cursor2y = cursor2y - 6.187927353e+28;

}

}

if (unused\_variable == 5) {

cursor2z = spawnerz + (1.856378206e+29 + 1.299464744e+28);

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(15 \* 1 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(12 \* 1 - 0);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_z();

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid\_wall\_z();

lipid2 = 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2y = cursor2y + 4.950341882e+28;

lipid2 = 4.950341882e+28;

lipid\_wall\_z();

lipid2 = 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2z = cursor2z - 3.588997865e+30;

lipid2 = 2.351412394e+30;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 4.950341882e+28;

lipid\_wall\_y();

cursor2z = cursor2z - 1.23758547e+28;

lipid1 = 4.950341882e+28;

lipid2 = 4.826583335e+28;

lipid\_wall\_x();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = Math.pow(((2.5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+28 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(4.950341882e+28 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(4.826583335e+28 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

salt\_calcium();

list\_molecules.push(['salt calcium', cursor1z, cursor1y, cursor1x, cursor1z + 1.92206873e+25, cursor1y - 1.92206873e+25, cursor1x - 1.92206873e+25]);

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2x = cursor2x - 1.23758547e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.23758547e+28;

cursor2y = cursor2y - 4.950341882e+28;

cursor2x = cursor2x + 4.950341882e+28;

lipid1 = 4.950341882e+28;

lipid2 = 1.23758547e+29;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2z = cursor2z + 3.588997865e+28;

lipid\_wall\_y();

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+28;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+28;

cursor2y = cursor2y - 1.23758547e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 1.23758547e+28;

cursor2z = cursor2z - 3.588997865e+28;

cursor2x = cursor2x - 4.950341882e+28;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.856378206e+29 + 1.299464744e+28);

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(15 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_y();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.23758547e+29 + 1.299464744e+28);

cursor2y = spawnery - (1.23758547e+28 + (2.351412394e+29 + 11 \* 6.187927353e+28));

cursor2x = spawnerx - 1.23758547e+29;

repeat\_endcell1 = Math.round(15 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(93 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_y();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_y();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = cursor2z - 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + (1.856378206e+29 + 1.299464744e+28);

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - (1.23758547e+29 - 4.950341882e+28);

repeat\_endcell1 = Math.round(93 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_x();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

cursor2z = spawnerz + (1.856378206e+29 + 1.299464744e+28);

cursor2y = spawnery - 2.351412394e+29;

cursor2x = spawnerx - (1.23758547e+29 + (1.23758547e+28 + 14 \* 6.187927353e+28));

repeat\_endcell1 = Math.round(93 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(16 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

lipid2 = 1.23758547e+28;

lipid1 = 1.23758547e+28;

lipid\_wall\_x();

cursor2z = cursor2z + 3.588997865e+30;

lipid\_wall\_x();

cursor2z = cursor2z - 3.588997865e+30;

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2z = cursor2z + 6.187927353e+28;

}

}

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - (1.856378206e+29 + 0);

cursor2x = spawnerx - (1.23758547e+29 + 1.23758547e+28);

repeat\_endcell1 = Math.round(15 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(11 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 4.331549147e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 4.331549147e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 200);

if (randomizer <= 50) {

peptidesequence = '1 mttqaptftq plqsvvvleg statfeahis gfpvpevswf rdgqvistst lpgvqisfsd 61 grakltipav tkansgrysl katngsgqat staellvkae tappnfvqrl qsmtvrqgsq 121 vrlqvrvtgi ptpvvkfyrd gaeiqssldf qisqegdlys lliaeayped sgtysvnatn 181 svgratstae llvqgeeevp akktktivst aqisesrqtr iekkieahfd arsiatvemv 241 idgaagqqlp hktpprippk pksrsptpps iaakaqlarq qspspirhsp spvrhvrapt 301 pspvrsvspa aristspirs vrspllmrkt qastvatgpe vpppwkqegy vassseaemr 361 ettlttstqi rteerwegry gvqeqvtisg aagaaasvsa sasyaaeava tgakevkqda 421 dksaavatvv aavdmarvre pvisaveqta qrttttavhi qpaqeqvrke aektavtkvv 481 vaadkakeqe lksrtkevit tkqeqmhvth eqirketekt fvpkvvisaa kakeqetris 541 eeitkkqkqv tqeairqete itaasmvvva takstkletv pgaqeetttq qdqmhlsyek 601 imketrktvv pkvivatpkv keqdlvsrgr egittkreqv qitqekmrke aektalstia 661 vatakakeqe tilrtretma trqeqiqvth gkvdvgkkae avatvvaavd qarvreprep 721 ghleesyaqq ttleygyker isaakvaepp qrpasephvv pkavkprviq apsethiktt 781 dqkgmhissq ikkttdltte rlvhvdkrpr tasphftvsk isvpktehgy easiagsaia 841 tlqkelsats saqkitksvk aptvkpsetr vraeptplpq fpfadtpdty kseagvevkk 901 evgvsitgtt vreerfevlh greakvteta rvpapveipv tpptlvsglk nvtviegesv 961 tlechisgyp sptvtwyred yqiessidfq itfqsgiarl mireafaeds grftcsavne 1021 agtvstscyl avqvseefek ettavtekft teekrfvesr dvvmtdtslt eeqagpgepa 1081 apyfitkpvv qklveggsvv fgcqvggnpk phvywkksgv plttgyrykv synkqtgeck 1141 lvismtfadd ageytivvrn khgetsasas lleeadyell mksqqemlyq tqvtafvqep 1201 kvgetapgfv yseyekeyek eqalirkkma kdtvvvrtyv edqefhissf eerlikeiey 1261 riikttleel leedgeekma vdiseseave sgfdsrikny rilegmgvtf hckmsgyplp 1321 kiawykdgkr ikhgeryqmd flqdgraslr ipvvlpedeg iytafasnik gnaicsgkly 1381 vepaaplgap tyiptlepvs rirslsprsv srspirmspa rmsparmspa rmsparmspg 1441 rrleetdesq lerlykpvfv lkpvsfkcle gqtarfdlkv vgrpmpetfw fhdgqqivnd 1501 ythkvviked gtqsliivpa tpsdsgewtv vaqnragrss isviltveav ehqvkpmfve 1561 klknvnikeg srlemkvrat gnpnpdivwl knsdiivphk ypkiriegtk geaalkidst 1621 vsqdsawyta tainkagrdt trckvnveve faepeperkl iiprgtyrak eiaapelepl 1681 hlrygqeqwe egdlydkekq qkpffkkklt slrlkrfgpa hfecrltpig dptmvvewlh 1741 dgkpleaanr lrminefgyc sldygvaysr dsgiitcrat nkygtdhtsa tlivkdeksl 1801 veesqlpegr kglqrieele rmahegaltg vttdqkekqk pdivlypepv rvlegetarf 1861 rcrvtgypqp kvnwylngql irkskrfrvr ydgihyldiv dcksydtgev kvtaenpegv 1921 iehkvkleiq qredfrsvlr rapeprpefh vhepgklqfe vqkvdrpvdt tetkevvklk 1981 raerithekv peeseelrsk fkrrteegyy eaitavelks rkkdesyeel lrktkdellh 2041 wtkelteeek kalaeegkit iptfkpdkie lspsmeapki feriqsqtvg qgsdahfrvr 2101 vvgkpdpece wykngvkier sdriywywpe dnvcelvird vtaedsasim vkainiaget 2161 sshafllvqa kqlitftqel qdvvakekdt matfecetse pfvkvkwykd gmevhegdky 2221 rmhsdrkvhf lsiltidtsd aedyscvlve denvkttakl ivegavvefv kelqdievpe 2281 sysgeleciv speniegkwy hndvelksng kytitsrrgr qnltvkdvtk edqgeysfvi 2341 dgkkttcklk mkprpiailq glsdqkvceg divqlevkvs lesvegvwmk dgqevqpsdr 2401 vhividkqsh mlliedmtke dagnysftip alglstsgrv svysvdvitp lkdvnviegt 2461 kavleckvsv pdvtsvkwyl ndeqikpddr vqaivkgtkq rlvinrthas degpyklivg 2521 rvetncnlsv ekikiirglr dltctetqnv vfevelshsg idvlwnfkdk eikpsskyki 2581 eahgkiyklt vlnmmkddeg kytfyagenm tsgkltvagg aiskpltdqt vaesqeavfe 2641 cevanpdskg ewlrdgkhlp ltnnirsesd ghkrrliiaa tklddigeyt ykvatsktsa 2701 klkveavkik ktlknltvte tqdavftvel thpnvkgvqw ikngvvlesn ekyaisvkgt 2761 iyslriknca ivdesvygfr lgrlgasarl hvetvkiikk pkdvtalena tvafevsvsh 2821 dtvpvkwfhk sveikpsdkh rlvserkvhk lmlqnispsd ageytavvgq leckaklfve 2881 tlhitktmkn ievpetktas fecevshfnv psmwlkngve iemsekfkiv vqgklhqlii 2941 mntstedsae ytfvcgndqv satltvtpim itsmlkdina eekdtitfev tvnyegisyk 3001 wlkngveiks tdkcqmrtkk lthslnirnv hfgdaadytf vagkatstat lyvearhief 3061 rkhikdikvl ekkramfece vsepditvqw mkddqelqit drikiqkeky vhrllipstr 3121 msdagkytvv aggnvstakl fvegrdvrir sikkevqvie kqravvefev neddvdahwy 3181 kdgieinfqv qerhkyvver rihrmfiset rqsdageytf vagrnrssvt lyvnapeppq 3241 vlqelqpvtv qsgkparfca visgrpqpki swykeeqlls tgfkckflhd gqeytlllie 3301 afpedaavyt ceakndygva ttsaslsvev pevvspdqem pvyppaiitp lqdtvtsegq 3361 parfqcrvsg tdlkvswysk dkkikpsrff rmtqfedtyq leiaeayped egtytfvasn 3421 avgqvsstan lsleapesil herieqeiem emkefsssfl saeeeglhsa elqlskinet 3481 lellsespvy ptkfdsekeg tgpifikevs nadismgdva tlsvtvigip kpkiqwffng 3541 vlltpsadyk fvfdgddhsl iilftklede geytcmasnd ygkticsayl kinskgeghk 3601 dtetesavak sleklggpcp phflkelkpi rcaqglpaif eytvvgepap tvtwfkenkq 3661 lctsvyytii hnpngsgtfi vndpqredsg lyickaenml gestcaaell vlledtdmtd 3721 tpckakstpe apedfpqtpl kgpavealds eqeiatfvkd tilkaalite enqqlsyehi 3781 akanelssql plgaqelqsi leqdkltpes treflcings ihfqplkeps pnlqlqivqs 3841 qktfskegil mpeepetqav lsdtekifps amsieqinsl tveplktlla epegnypqss 3901 ieppmhsylt svaeevlspk ektvsdtnre qrvtlqkqea qsalilsqsl aeghveslqs 3961 pdvmisqvny eplvpsehsc teggkilies anplenagqd savrieegks lrfplaleek 4021 qvllkeehsd nvvmppdqii eskrepvaik kvqevqgrdl lskesllsgi peeqrlnlki 4081 qicralqaav aseqpglfse wlrniekvev eavnitqepr himcmylvts aksvteevti 4141 iiedvdpqma nlkmelrdal caiiyeeidi ltaegpriqq gaktslqeem dsfsgsqkve 4201 pitepevesk ylisteevsy fnvqsrvkyl datpvtkgva savvsdekqd eslkpseeke 4261 esssesgtee vatvkiqeae gglikedgpm ihtplvdtvs eegdivhltt sitnakevnw 4321 yfenklvpsd ekfkclqdqn tytlvidkvn tedhqgeyvc ealndsgkta tsakltvvkr 4381 aapvikrkie plevalghla kftceiqsap nvrfqwfkag reiyesdkcs irsskyissl 4441 eilrtqvvdc geytckasne ygsvsctatl tvteaypptf lsrpkslttf vgkaakfict 4501 vtgtpvieti wqkdgaalsp spnwrisdae nkhilelsnl tiqdrgvysc kasnkfgadi 4561 cqaeliiidk phfikelepv qsainkkvhl ecqvdedrkv tvtwskdgqk lppgkdykic 4621 fedkiatlei plaklkdsgt yvctasneag ssscsatvtv reppsfvkkv dpsylmlpge 4681 sarlhcklkg spviqvtwfk nnkelsesnt vrmyfvnsea ilditdvkve dsgsysceav 4741 ndvgsdscst eivikeppsf iktlepadiv rgtnallqce vsgtgpfeis wfkdkkqirs 4801 skkyrlfsqk slvcleifsf nsadvgeyec vvanevgkcg cmathllkep ptfvkkvddl 4861 ialggqtvtl qaavrgsepi svtwmkgqev iredgkikms fsngvavlii pdvqisfggk 4921 ytclaeneag sqtsvgeliv kepakiiera eliqvtagdp atleytvagt pelkpkwykd 4981 grplvaskky risfknnvaq lkfysaelhd sgqytfeisn evgssscett ftvldrdiap 5041 fftkplrnvd svvngtcrld ckiagslpmr vswfkdgkei aasdryriaf vegtasleii 5101 rvdmndagnf tcratnsvgs kdssgalivq eppsfvtkpg skdvlpgsav clkstfqgst 5161 pltirwfkgn kelvsggscy itkealessl elylvktsds gtytckvsnv aggvecsanl 5221 fvkepatfve klepsqllkk gdatqlackv tgtppikitw fandreikes skhrmsfves 5281 tavlrltdvg iedsgeymce aqneagsdhc ssivivkesp yftkefkpie vlkeydvmll 5341 aevagtppfe itwfkdntil rsgrkyktfi qdhlvslqil kfvaadagey qcrvtnevgs 5401 sicsarvtlr eppsfikkie stsslrggta afqatlkgsl pitvtwlkds deiteddnir 5461 mtfennvasl ylsgievkhd gkyvcqaknd agiqrcsall svkepatite eavsidvtqg 5521 dpatlqvkfs gtkeitakwf kdgqeltlgs kykisvtdtv silkiistek kdsgeytfev 5581 qndvgrssck arinvldlii ppsftkklkk mdsikgsfid lecivagshp isiqwfkddq 5641 eisasekykf sfhdntafle isqlegtdsg tytcsatnka ghnqcsghlt vkeppyfvek 5701 pqsqdvnpnt rvqlkalvgg tapmtikwfk dnkelhsgaa rsvwkddtst slelfaakat 5761 dsgtyicqls ndvgtatska tlfvkeppqf ikkpspvlvl rngqsttfec qitgtpkirv 5821 swyldgneit aiqkhgisfi dglatfqisg arvensgtyv cearndagta scsielkvke 5881 pptfirelkp vevvkysdve lecevtgtpp fevtwlknnr eirsskkytl tdrvsvfnlh 5941 itkcdpsdtg eyqcivsneg gscscstrva lkeppsfikk ientttvlks satfqstvag 6001 sppisitwlk ddqildeddn vyisfvdsva tlqirsvdng hsgrytcqak nesgvercya 6061 fllvqepaqi vekaksvdvt ekdpmtlecv vagtpelkvk wlkdgkqivp sryfsmsfen 6121 nvasfriqsv mkqdsgqytf kvendfgsss cdaylrvldq nippsftkkl tkmdkvlgss 6181 ihmeckvsgs lpisaqwfkd gkeistsaky rlvchersvs levnnleled tanytckvsn 6241 vagddacsgi ltvkeppsfl vkpgrqqaip dstvefkail kgtppfkikw fkddvelvsg 6301 pkcfiglegs tsflnlysvd asktgqytch vtndvgsdsc ttmllvtepp kfvkkleask 6361 ivkagdssrl eckiagspei rvvwfrnehe lpasdkyrmt fidsvaviqm nnlstedsgd 6421 ficeaqnpag stscstkviv keppvfssfp pivetlknae vslecelsgt ppfevvwykd 6481 krqlrsskky kiasknfhts ihilnvdtsd igeyhckaqn evgsdtcvct vklkepprfv 6541 sklnsltvva gepaelqasi egaqpifvqw lkekeevire seniritfve nvatlqfaka 6601 epanagkyic qikndggmre nmatlmvlep avivekagpm tvtvgetctl eckvagtpel 6661 svewykdgkl ltssqkhkfs fynkisslri lsverqdagt ytfqvqnnvg kssctavvdv 6721 sdravppsft rrlkntggvl gascileckv agsspisvaw fhektkivsg akyqttfsdn 6781 vctlqlnsld ssdmgnytcv aanvagsdec ravltvqepp sfvkepeple vlpgknvtft 6841 svirgtppfk vnwfrgarel vkgdrcniyf edtvaelelf nidisqsgey tcvvsnnagq 6901 ascttrlfvk epaaflkrls dhsvepgksi ilestytgtl pisvtwkkdg fnittsekcn 6961 ivttektcil eilnstkrda gqysceiene agrdvcgalv stleppyfvt elepleaavg 7021 dsvslqcqva gtpeitvswy kgdtklrptp eyrtyftnnv atlvfnkvni ndsgeytcka 7081 ensigtassk tvfriqerql ppsfarqlkd ieqtvglpvt ltcrlngsap iqvcwyrdgv 7141 llrddenlqt sfvdnvatlk ilqtdlshsg qyscsasnpl gtasssarlt arepkkspff 7201 dikpvsidvi agesadfech vtgaqpmrit wskdnkeirp ggnytitcvg ntphlrilkv 7261 gkgdsgqytc qatndvgkdm csaqlsvkep pkfvkkleas kvakqgesiq leckisgspe 7321 ikvswfrnds elheswkynm sfinsvallt ineasaedsg dyiceahngv gdascstalt 7381 vkappvftqk pspvgalkgs dvilqceisg tppfevvwvk drkqvrnskk fkitskhfdt 7441 slhilnleas dvgeyhckat nevgsdtcsc svkfkepprf vkklsdtstl igdavelrai 7501 vegfqpisvv wlkdrgevir esentrisfi dniatlqlgs peasnsgkyi cqikndagmr 7561 ecsavltvle pariiekpep mtvttgnpfa lecvvtgtpe lsakwfkdgr elsadskhhi 7621 tfinkvaslk ipcaemsdkg lysfevknsv gksnctvsvh vsdrivppsf irklkdvnai 7681 lgasvvlecr vsgsapisvg wfqdgneivs gpkcqssfse nvctlnlsll epsdtgiytc 7741 vaanvagsde csavltvqep psfeqtpdsv evlpgmsltf tsvirgtppf kvkwfkgsre 7801 lvpgescnis ledfvtelel fevqplesgd ysclvtndag sasctthlfv kepatfvkrl 7861 adfsvetgsp ivleatytgt ppisvswikd eylisqserc sitmteksti leilestied 7921 yaqysclien eagqdiceal vsvleppyfi eplehveavi gepatlqckv dgtpeirisw 7981 ykehtklrsa paykmqfknn vaslvinkvd hsdvgeysck adnsvgavas savlvikark 8041 lppffarklk dvhetlgfpv afecringse plqvswykdg vllkddanlq tsfvhnvatl 8101 qilqtdqshi gqyncsasnp lgtasssakl ilsehevppf fdlkpvsvdl algesgtfkc 8161 hvtgtapiki twakdnreir pggnykmtlv entatltvlk vgkgdagqyt cyasniagkd 8221 scsaqlgvqe pprfikklep srivkqdeft ryeckiggsp eikvlwykde teiqesskfr 8281 msfvdsvavl emhnlsveds gdytceahna agsassstsl kvkeppifrk kphpietlkg 8341 advhlecelq gtppfhvswy kdkrelrsgk kykimsenfl tsihilnvda adigeyqcka 8401 tndvgsdtcv gsialkappr fvkklsdist vvgkevqlqt tiegaepisv vwfkdkgeiv 8461 resdniwisy seniatlqfs rvepanagky tcqikndagm qecfatlsvl epativekpe 8521 sikvttgdtc tlectvagtp elstkwfkdg keltsdnkyk isffnkvsgl kiinvapsds 8581 gvysfevqnp vgkdsctasl qvsdrtvpps ftrklketng lsgssvvmec kvygsppisv 8641 swfhegneis sgrkyqttlt dntcaltvnm leesdsgdyt ciatnmagsd ecsapltvre 8701 ppsfvqkpdp mdvltgtnvt ftsivkgtpp fsvswfkgss elvpgdrcnv sledsvaele 8761 lfdvdtsqsg eytcivsnea gkasctthly ikapakfvkr lndysiekgk plilegtftg 8821 tppisvtwkk nginvtpsqr cnittteksa ileipsstve dagqyncyie nasgkdscsa 8881 qilileppyf vkqlepvkvs vgdsaslqcq lagtpeigvs wykgdtklrp tttykmhfrn 8941 nvatlvfnqv dindsgeyic kaensvgevs astfltvqeq klppsfsrql rdvqetvglp 9001 vvfdcaisgs episvswykd gkplkdspnv qtsfldntat lnifktdrsl agqysctatn 9061 pigsasssar liltegknpp ffdirlapvd avvgesadfe chvtgtqpik vswakdsrei 9121 rsggkyqisy lensahltvl kvdkgdsgqy tcyavnevgk dsctaqlnik erlippsftk 9181 rlsetveete gnsfklegrv agsqpitvaw yknnieiqpt snceitfknn tlvlqvrkag 9241 mndaglytck vsndagsalc tssivikepk kppvfdqhlt pvtvsegeyv qlschvqgse 9301 piriqwlkag reikpsdrcs fsfasgtavl elrdvakads gdyvckasnv agsdttkskv 9361 tikdkpavap atkkaavdgr lffvsepqsi rvvekttatf iakvggdpip nvkwtkgkwr 9421 qlnqggrvfi hqkgdeakle irdttktdsg lyrcvafneh geiesnvnlq vderkkqeki 9481 egdlramlkk tpilkkgage eeeidimell knvdpkeyek yarmygitdf rgllqafell 9541 kqsqeeethr leieeierse rdekefeelv sfiqqrlsqt epvtlikdie nqtvlkdnda 9601 vfeidikiny peiklswykg teklepsdkf eisidgdrht lrvkncqlkd qgnyrlvcgp 9661 hiasakltvi epawerhlqd vtlkegqtct mtcqfsvpnv ksewfrngri lkpqgrhkte 9721 vehkvhklti advraedqgq ytckyedlet saelrieaep iqftkriqni vvsehqsatf 9781 ecevsfddai vtwykgptel tesqkynfrn dgrchymtih nvtpddegvy sviarleprg 9841 earstaelyl ttkeiklelk ppdipdsrvp iptmpiravp peeippvvap piplllptpe 9901 ekkpppkrie vtkkavkkda kkvvakpkem tpreeivkkp pppttlipak apeiidvssk 9961 aeevkimtit rkkevqkeke avyekkqavh kekrvfiesf eepydeleve pytepfeqpy 10021 yeepdedyee ikveakkevh eeweedfeeg qeyyereegy degeeeweea yqereviqvq 10081 kevyeesher kvpakvpekk appppkvikk pviekiekts rrmeeekvqv tkvpevskki 10141 vpqkpsrtpv qeevievkvp avhtkkmvis eekmffasht eeevsvtvpe vqkeivteek 10201 ihvaiskrve pppkvpelpe kpapeevapv pipkkveppa pkvpevpkkp vpeekkpvpv 10261 pkkepaappk vpevpkkpvp eekipvpvak kkeappakvp evqkgvvtee kitivtqree 10321 spppavpeip kkkvpeerkp vprkeeevpp ppkvpalpkk pvpeekvavp vpvakkappp 10381 raevskktvv eekrfvaeek lsfavpqrve vtrhevsaee ewsyseeeeg vsisvyreee 10441 reeeeeaevt eyevmeepee yvveeklhii skrveaepae vterqekkiv lkpkipakie 10501 epppakvpea pkkivpekkv papvpkkekv pppkvpeepk kpvpekkvpp kvikmeeplp 10561 akvterhmqi tqeekvlvav tkkeappkar vpeepkravp eekvlklkpk reeeppakvt 10621 efrkrvvkee kvsieapkre pqpikevtim eekeraytle eeavsvqree eyeeyeeydy 10681 kefeeyepte eydqyeeyee reyeryeehe eyitepekpi pvkpvpeepv ptkpkappak 10741 vlkkavpeek vpvpipkklk ppppkvpeep kkvfeekiri sitkrekeqv tepaakvpmk 10801 pkrvvaeekv pvprkevapp vrvpevpkel epeevafeee vvthveeylv eeeeeyihee 10861 eefiteeevv pvipvkvpev prkpvpeekk pvpvpkkkea ppakvpevpk kpeekvpvli 10921 pkkekpppak vpevpkkpvp eekvpvpvpk kveappakvp evpkkpvpek kvpvpapkkv 10981 eappakvpev pkklipeekk ptpvpkkvea pppkvpkkre pvpvpvalpq eeevlfeeei 11041 vpeeevlpee eevlpeeeev lpeeeevlpe eeeippeeee vppeeeyvpe eeefvpeeev 11101 lpevkpkvpv papvpeikkk vtekkvvipk keeappakvp evpkkveekr iilpkeeevl 11161 pvevteepee episeeeipe eppsieevee vapprvpevi kkavpeaptp vpkkveappa 11221 kvskkipeek vpvpvqkkea ppakvpevpk kvpekkvlvp kkeavppakg rtvleekvsv 11281 afrqevvvke rlelevveae veeipeeeef heveeyfeeg efheveefik leqhrveeeh 11341 rvekvhrvie vfeaeevevf ekpkappkgp eisekiippk kpptkvvprk eppakvpevp 11401 kkivveekvr vpeeprvppt kvpevlppke vvpekkvpvp pakkpeappp kvpeapkevv 11461 pekkvpvppp kkpevpptkv pevpkaavpe kkvpeaippk pespppevpe apkevvpekk 11521 vpaappkkpe vtpvkvpeap kevvpekkvp vpppkkpevp ptkvpevpkv avpekkvpea 11581 ippkpesppp evfeepeeva leeppaevve epepaappqv tvppkkpvpe kkapavvakk 11641 pelppvkvpe vpkevvpekk vplvvpkkpe appakvpevp kevvpekkva vpkkpevppa 11701 kvpevpkkpv leekpavpvp eraespppev yeepeeiape eeiapeeekp vpvaeeeepe 11761 vpppavpeep kkiipekkvp vikkpeappp kepepekvie kpklkprppp pppappkedv 11821 kekifqlkai pkkkvpekpq vpekveltpl kvpggekkvr kllperkpep keevvlksvl 11881 rkrpeeeepk vepkklekvk kpavpepppp kpveevevpt vtkrerkipe ptkvpeikpa 11941 iplpapepkp kpeaevktik pppvepeptp iaapvtvpvv gkkaeakapk eeaakpkgpi 12001 kgvpkktpsp ieaerrklrp gsggekppde apftyqlkav plkfvkeikd iiltesefvg 12061 ssaifeclvs pstaittwmk dgsnirespk hrfiadgkdr klhiidvqls dageytcvlr 12121 lgnkektsta klvveelpvr fvktleeevt vvkgqplyls celnkerdvv wrkdgkivve 12181 kpgrivpgvi glmraltind addtdagtyt vtvenannle csscvkvvev irdwlvkpir 12241 dqhvkpkgta ifacdiakdt pnikwfkgyd eipaepndkt eilrdgnhly lkiknamped 12301 iaeyaveieg krypakltlg erevellkpi edvtiyekes asfdaeisea dipgqwklkg 12361 ellrpsptce ikaeggkrfl tlhkvkldqa gevlyqalna ittailtvke ieldfavplk 12421 dvtvperrqa rfecvltrea nviwskgpdi ikssdkfdii adgkkhilvi ndsqfddegv 12481 ytaevegkkt sarlfvtgir lkfmspledq tvkegetatf vcelshekmh vvwfkndakl 12541 htsrtvliss egkthklemk evtlddisqi kaqvkelsst aqlkvleadp yftvklhdkt 12601 avekdeitlk cevskdvpvk wfkdgeeivp spkysikadg lrrilkikka dlkdkgeyvc 12661 dcgtdktkan vtvearlikv ekplygvevf vgetahfeie lsepdvhgqw klkgqpltas 12721 pdceiiedgk khililhncq lgmtgevsfq aanaksaanl kvkelplifi tplsdvkvfe 12781 kdeakfecev srepktfrwl kgtqeitgdd rfelikdgtk hsmviksaaf edeakymfea 12841 edkhtsgkli iegirlkflt plkdvtakek esavftvels hdnirvkwfk ndqrlhttrs 12901 vsmqdegkth sitfkdlsid dtsqirveam gmsseakltv legdpyftgk lqdytgvekd 12961 evilqceisk adapvkwfkd gkeikpskna vikadgkkrm lilkkalksd igqytcdcgt 13021 dktsgkldie dreiklvrpl hsvevmetet arfeteised dihanwklkg eallqtpdce 13081 ikeegkihsl vlhncrldqt ggvdfqaanv kssahlrvkp rvigllrplk dvtvtageta 13141 tfdcelsyed ipvewylkgk klepsdkvvp rsegkvhtlt lrdvkledag evqltakdfk 13201 thanlfvkep pveftkpled qtveegatav lecevsrena kvkwfkngte ilkskkyeiv 13261 adgrvrklvi hdctpedikt ytcdakdfkt scnlnvvpph veflrpltdl qvrekemarf 13321 ecelsrenak vkwfkdgaei kkgkkydiis kgavrilvin kcllddeaey scevrtarts 13381 gmltvleeea vftknlanie vsetdtiklv cevskpgaev iwykgdeeii etgryeilte 13441 grkrilviqn ahledagnyn crlpssrtdg kvkvhelaae fiskpqnlei legekaefvc 13501 siskesfpvq wkrddktles gdkydviadg kkrvlvvkda tlqdmgtyvv mvgaaraaah 13561 ltvieklriv vplkdtrvke qqevvfncev ntegakakwf rneeaifdss kyiilqkdlv 13621 ytlrirdahl ddqanynvsl tnhrgenvks aanliveeed lriveplkdi etmekksvtf 13681 wckvnrlnvt lkwtkngeev pfdnrvsyrv dkykhmltik dcgfpdegey ivtagqdksv 13741 aelliieapt efvehledqt vtefddavfs cqlsrekanv kwyrngreik egkkykfekd 13801 gsihrliikd crlddeceya cgvedrksra rlfveeipve iirppqdile apgadvvfla 13861 elnkdkvevq wlrnnmvvvq gdkhqmmseg kihrlqicdi kprdqgeyrf iakdkearak 13921 lelaaapkik tadqdlvvdv gkpltmvvpy daypkaeaew fkeneplstk tidttaeqts 13981 frileakkgd kgrykivlqn khgkaegfin lkvidvpgpv rnlevtetfd gevslaweep 14041 ltdggskiig yvverrdikr ktwvlatdra esceftvtgl qkggveylfr vsarnrvgtg 14101 epvetdnpve arskydvpgp plnvtitdvn rfgvsltwep peydggaeit nyvielrdkt 14161 sirwdtamtv raedlsatvt dvvegqeysf rvraqnrigv gkpsaatpfv kvadpierps 14221 ppvnltssdq tqssvqlkwe pplkdggspi lgyiiercee gkdnwircnm klvpeltykv 14281 tglekgnkyl yrvsaenkag vsdpseilgp ltaddafvep tmdlsafkdg levivpnpit 14341 ilvpstgypr ptatwcfgdk vletgdrvkm ktlsayaelv ispsersdkg iytlklenrv 14401 ktisgeidvn viarpsapke lkfgditkds vhltweppdd dggspltgyv vekrevsrkt 14461 wtkvmdfvtd leftvpdlvq gkeylfkvca rnkcgpgepa yvdepvnmst patvpdppen 14521 vkwrdrtans ifltwdppkn dggsrikgyi vercprgsdk wvacgepvae tkmevtglee 14581 gkwyayrvka lnrqgaskps rpteeiqavd tqeapeifld vkllagltvk agtkielpat 14641 vtgkpepkit wtkadmilkq dkritienvp kkstvtivds krsdtgtyii eavnvcgrat 14701 avvevnvldk pgppaafdit dvtnescllt wnpprddggs kitnyvverr atdsevwhkl 14761 sstvkdtnfk atklipnkey ifrvaaenmy gvgepvqasp itakyqfdpp gpptrlepsd 14821 itkdavtltw cepdddggsp itgywverld pdtdkwvrcn kmpvkdttyr vkgltnkkky 14881 rfrvlaenla gpgkpskste pilikdpidp pwppgkptvk dvgktsvrln wtkpehdgga 14941 kiesyvieml ktgtdewvrv aegvpttqhl lpglmegqey sfrvravnka gesepsepsd 15001 pvlcreklyp pspprwlevi nitkntadlk wtvpekdggs pitnyivekr dvrrkgwqtv 15061 dttvkdtkct vtpltegsly vfrvaaenai gqsdyteied svlakdtftt pgppyalavv 15121 dvtkrhvdlk weppkndggr piqryviekk erlgtrwvka gktagpdcnf rvtdviegte 15181 vqfqvraene agvghpsept eilsiedpts ppsppldlhv tdagrkhiai awkppekngg 15241 spiigyhvem cpvgtekwmr vnsrpikdlk fkveegvvpd keyvlrvrav naigvsepse 15301 isenvvakdp dckptidlet hdiiviegek lsipvpfrav pvptvswhkd gkevkasdrl 15361 tmkndhisah levpksvrad agiytitlen klgsatasin vkviglpgpc kdikasditk 15421 ssckltwepp efdggtpilh yvlerreagr rtyipvmsge nklswtvkdl ipngeyffrv 15481 kavnkvggge yielknpvia qdpkqppdpp vdvevhnpta eamtitwkpp lydggskimg 15541 yiiekiakge erwkrcnehl vpiltytakg leegkeyqfr vraenaagis epsratpptk 15601 avdpidapkv ilrtslevkr gdeialdasi sgspyptitw ikdenvivpe eikkraaplv 15661 rrrkgevqee epfvlpltqr lsidnskkge sqlrvrdslr pdhglymikv endhgiakap 15721 ctvsvldtpg ppinfvfedi rktsvlckwe pplddggsei inytlekkdk tkpdsewivv 15781 tstlrhckys vtkliegkey lfrvraenrf gpgppcvskp lvakdpfgpp dapdkpived 15841 vtsnsmlvkw nepkdngspi lgywlekrev nsthwsrvnk sllnalkanv dgllegltyv 15901 frvcaenaag pgkfsppsdp ktahdpispp gppiprvtdt ssttielewe ppafngggei 15961 vgyfvdkqlv gtnewsrcte kmikvrqytv keiregadyk lrvsavnaag egppgetqpv 16021 tvaepqeppa veldvsvkgg iqimagktlr ipavvtgrpv ptkvwtkeeg eldkdrvvid 16081 nvgtkselii kdalrkdhgr yvitatnscg skfaaarvev fdvpgpvldl kpvvtnrkmc 16141 llnwsdpedd ggseitgfii erkdakmhtw rqpietersk cditgllegq eykfrviakn 16201 kfgcgppvei gpilavdplg pptsperlty tertkstitl dwkeprsngg spiqgyiiek 16261 rrhdkpdfer vnkrlcptts flvenldehq myefrvkavn eigesepslp lnvviqddev 16321 pptiklrlsv rgdtikvkag epvhipadvt glpmpkiews knetviekpt dalqitkeev 16381 srseaktels ipkavredkg tytvtasnrl gsvfrnvhve vydrpspprn lavtdikaes 16441 cyltwdapld nggseithyv idkrdasrkk aeweevtnta vekrygiwkl ipngqyefrv 16501 ravnkygisd ecksdkvviq dpyrlpgppg kpkvlartkg smlvswtppl dnggspitgy 16561 wlekreegsp ywsrvsrapi tkvglkgvef nvprllegvk yqframaina agigppseps 16621 dpevagdpif ppgppscpev kdktkssisl gwkppakdgg spikgyivem qeegttdwkr 16681 vnepdklitt cecvvpnlke lrkyrfrvka vneageseps dttgeipatd iqeepevfid 16741 igaqdclvck agsqiripav ikgrptpkss wefdgkakka mkdgvhdipe daqletaens 16801 sviiipeckr shtgkysita knkagqktan crvkvmdvpg ppkdlkvsdi trgscrlswk 16861 mpdddggdri kgyviekrti dgkawtkvnp dcgsttfvvp dllseqqyff rvraenrfgi 16921 gppvetiqrt tardpiyppd ppiklkigli tkntvhlswk ppkndggspv thyiveclaw 16981 dptgtkkeaw rqcnkrdvee lqftvedlve ggeyefrvka vnaagvskps atvgpvtvkd 17041 qtcppsidlk efmeveegtn vnivakikgv pfptltwfka ppkkpdnkep vlydthvnkl 17101 vvddtctlvi pqsrrsdtgl ytitavnnlg taskemrlnv lgrpgppvgp ikfesvsadq 17161 mtlswfppkd dggskitnyv iekreanrkt wvhvssepke ctytipklle gheyvfrima 17221 qnkygigepl dsepetarnl fsvpgapdkp tvssvtrnsm tvnweepeyd ggspvtgywl 17281 emkdttskrw krvnrdpika mtlgvsykvt gliegsdyqf rvyainaagv gpaslpsdpa 17341 tardpiappg ppfpkvtdwt kssadlewsp plkdggskvt gyiveykeeg keewekgkdk 17401 evrgtklvvt glkegafykf rvravniagi gepgevtdvi emkdrlvspd lqldasvrdr 17461 ivvhaggvir iiayvsgkpp ptvtwnmner tlpqeatiet taisssmvik ncqrshqgvy 17521 sllakneage rkktiivdvl dvpgpvgtpf lahnltnesc kltwfspedd ggspitnyvi 17581 ekresdrraw tpvtytvtrq natvqgliqg kayffriaae nsigmgpfve tsealvirep 17641 itvperpedl evkevtkntv tltwnppkyd ggseiinyvl esrligtekf hkvtndnlls 17701 rkytvkglke gdtyeyrvsa vnivgqgkps fctkpitckd elapptlhld frdkltirvg 17761 eafaltgrys gkpkpkvswf kdeadvledd rthikttpat lalekikakr sdsgkycvvv 17821 enstgsrkgf cqvnvvdrpg ppvgpvsfde vtkdymvisw kpplddggsk itnyiiekke 17881 vgkdvwmpvt sasakttckv skllegkdyi frihaenlyg isdplvsdsm kakdrfrvpd 17941 apdqpivtev tkdsalvtwn kphdggkpit nyilekretm skrwarvtkd pihpytkfrv 18001 pdllegcqye frvsaeneig igdpsppskp vfakdpiakp sppvnpeaid ttcnsvdltw 18061 qpprhdggsk ilgyiveyqk vgdeewrran htpescpetk ykvtglrdgq tykfrvlavn 18121 aagesdpahv pepvlvkdrl eppelildan mareqhikvg dtlrlsaiik gvpfpkvtwk 18181 kedrdaptka ridvtpvgsk leirnaahed ggiysltven pagsktvsvk vlvldkpgpp 18241 rdlevseirk dscyltwkep lddggsvitn yvverrdvas aqwsplsats kkkshfakhl 18301 negnqylfrv aaenqygrgp fvetpkpika ldplhppgpp kdlhhvdvdk tevslvwnkp 18361 drdggspitg ylveyqeegt qdwikfktvt nlecvvtglq qgktyrfrvk aenivglglp 18421 dttipiecqe klvppsveld vklieglvvk agttvrfpai irgvpvptak wttdgseikt 18481 dehytvetdn fssvltiknc lrrdtgeyqi tvsnaagskt vavhltvldv pgpptgpini 18541 ldvtpehmti swqppkddgg spvinyivek qdtrkdtwgv vssgssktkl kiphlqkgce 18601 yvfrvraenk igvgppldst ptvakhkfsp psppgkpvvt ditenaatvs wtlpksdggs 18661 pitgyymerr evtgkwvrvn ktpiadlkfr vtglyegnty efrvfaenla glskpspssd 18721 pikacrpikp pgppinpklk dksretadlv wtkplsdggs pilgyvvecq kpgtaqwnri 18781 nkdelirqca frvpgliegn eyrfrikaan ivgegeprel aesviakdil hppeveldvt 18841 crdvitvrvg qtirilarvk grpepditwt kegkvlvrek rvdliqdlpr velqikeavr 18901 adhgkyiisa knssghaqgs aivnvldrpg pcqnlkvtnv tkenctiswe npldnggsei 18961 tnfiveyrkp nqkgwsivas dvtkrlikan llanneyyfr vcaenkvgvg ptietktpil 19021 ainpidrpge penlhiadkg ktfvylkwrr pdydggspnl syhverrlkg sddwervhkg 19081 sikethymvd rcvenqiyef rvqtknegge sdwvkteevv vkedlqkpvl dlklsgvltv 19141 kagdtirlea gvrgkpfpev awtkdkdatd ltrsprvkid tradsskfsl tkakrsdggk 19201 yvvtatntag sfvayatvnv ldkpgpvrnl kivdvssdrc tvcwdppedd ggceiqnyil 19261 ekcetkrmvw stysatvltp gttvtrlieg neyifrvrae nkigtgppte skpviaktky 19321 dkpgrpdppe vtkvskeemt vvwnppeydg gksitgyfle kkekhstrwv pvnksaiper 19381 rmkvqnllpd heyqfrvkae neigigepsl psrpvvakdp ieppgpptnf rvvdttkhsi 19441 tlgwgkpvyd ggapiigyvv emrpkiadas pdegwkrcna aaqlvrkeft vtsldenqey 19501 efrvcaqnqv gigrpaelke aikpkeilep peidldasmr klvivragcp irlfaivrgr 19561 papkvtwrkv gidnvvrkgq vdlvdtmafl vipnstrdds gkysltlvnp agekavfvnv 19621 rvldtpgpvs dlkvsdvtkt schvswappe ndggsqvthy ivekreadrk twstvtpevk 19681 ktsfhvtnlv pgneyyfrvt avneygpgvp tdvpkpvlas dplsepdppr klevtemtkn 19741 satlawlppl rdggakidgy itsyreeeqp adrwteysvv kdlslvvtgl kegkkykfrv 19801 aarnavgvsl preaegvyea keqllppkil mpeqitikag kklrieahvy gkphptckwk 19861 kgedevvtss hlavhkadss siliikdvtr kdsgyyslta enssgtdtqk ikvvvmdapg 19921 ppqppfdisd idadacslsw hipledggsn itnyivekcd vsrgdwvtal asvtktscrv 19981 gklipgqeyi frvraenrfg isepltspkm vaqfpfgvps epknarvtkv nkdcifvawd 20041 rpdsdggspi igylierker nsllwvkand tlvrsteypc aglvegleys friyalnkag 20101 ssppskptey vtarmpvdpp gkpevidvtk stvsliwarp khdggskiig yfveacklpg 20161 dkwvrcntap hqipqeeyta tgleekaqyq fraiartavn isppsepsdp vtilaenvpp 20221 ridlsvamks lltvkagtnv cldatvfgkp mptvswkkdg tllkpaegik mamqrnlctl 20281 elfsvnrkds gdytitaens sgsksatikl kvldkpgppa svkinkmysd ramlsweppl 20341 edggseitny ivdkretsrp nwaqvsatvp itscsvekli egheyqfric aenkygvgdp 20401 vftepaiakn pydppgrcdp pvisnitkdh mtvswkppad dggspitgyl lekretqavn 20461 wtkvnrkpii ertlkatglq egteyefrvt ainkagpgkp sdaskaayar dpqyppgppa 20521 fpkvydttrs svslswgkpa ydggspiigy lvevkradsd nwvrcnlpqn lqktrfevtg 20581 lmedtqyqfr vyavnkigys dpsdvpdkhy pkdilippeg eldadlrktl ilragvtmrl 20641 yvpvkgrppp kitwskpnvn lrdrigldik stdfdtflrc envnkydagk yiltlenscg 20701 kkeytivvkv ldtpgppvnv tvkeiskdsa yvtweppiid ggspiinyvv qkrdaerksw 20761 stvttecskt sfrvanleeg ksyffrvfae neygigdpge trdavkasqt pgpvvdlkvr 20821 svsksscsig wkkphsdggs riigyvvdfl teenkwqrvm kslslqysak dltegkeytf 20881 rvsaenenge gtpseitvva rddvvapdld lkglpdlcyl akensnfrlk ipikgkpaps 20941 vswkkgedpl atdtrvsves savnttlivy dcqksdagky titlknvagt kegtisikvv 21001 gkpgiptgpi kfdevtaeam tlkwappkdd ggseitnyil ekrdsvnnkw vtcasavqkt 21061 tfrvtrlheg meytfrvsae nkygvgeglk sepivarhpf dvpdappppn ivdvrhdsvs 21121 ltwtdpkktg gspitgyhle fkernsllwk ranktpirmr dfkvtglteg leyefrvmai 21181 nlagvgkpsl psepvvaldp idppgkpevi nitrnsvtli wtepkydggh kltgyivekr 21241 dlpskswmka nhvnvpecaf tvtdlveggk yefriraknt agaisapses tetiickdey 21301 eaptivldpt ikdgltikag dtivlnaisi lgkplpkssw skagkdirps ditqitstpt 21361 ssmltikyat rkdageytit atnpfgtkve hvkvtvldvp gppgpveisn vsaekatltw 21421 tppledggsp iksyilekre tsrllwtvvs ediqscrhva tkliqgneyi frvsavnhyg 21481 kgepvqsepv kmvdrfgppg ppekpevsnv tkntatvswk rpvddggsei tgyhverrek 21541 kslrwvraik tpvsdlrckv tglqegstye frvsaenrag igppseasds vlmkdaaypp 21601 gppsnphvtd ttkksaslaw gkphydggle itgyvvehqk vgdeawikdt tgtalritqf 21661 vvpdlqtkek ynfrisaind agvgepavip dveiverema pdfeldaelr rtlvvragls 21721 irifvpikgr papevtwtkd ninlknrani entesftlli ipecnrydtg kfvmtienpa 21781 gkksgfvnvr vldtpgpvln lrptditkds vtlhwdlpli dggsritnyi vekreatrks 21841 ystattkchk ctykvtglse gceyffrvma eneygigept ettepvkase apsppdslni 21901 mditkstvsl awpkpkhdgg skitgyviea qrkgsdqwth ittvkglecv vrnltegeey 21961 tfqvmavnsa grsapresrp vivkeqtmlp eldlrgiyqk lviakagdni kveipvlgrp 22021 kptvtwkkgd qilkqtqrvn fettatstil ninecvrsds gpypltarni vgevgdviti 22081 qvhdipgppt gpikfdevss dfvtfswdpp endggvpisn yvvemrqtds ttwvelattv 22141 irttykatrl ttgleyqfrv kaqnrygvgp gitsacivan ypfkvpgppg tpqvtavtkd 22201 smtiswhepl sdggspilgy hverkerngi lwqtvskalv pgnifkssgl tdgiayefrv 22261 iaenmagksk pskpsepmla ldpidppgkp vplnitrhtv tlkwakpeyt ggfkitsyiv 22321 ekrdlpngrw lkanfsnile neftvsglte daayefrvia knaagaispp sepsdaitcr 22381 ddveapkikv dvkfkdtvil kageafrlea dvsgrppptm ewskdgkele gtakleikia 22441 dfstnlvnkd strrdsgayt ltatnpggfa khifnvkvld rpgppegpla vtevtsekcv 22501 lswfpplddg gakidhyivq kretsrlawt nvasevqvtk lkvtkllkgn eyifrvmavn 22561 kygvgeples epvlavnpyg ppdppknpev ttitkdsmvv cwghpdsdgg seiinyiver 22621 rdkagqrwik cnkktltdlr ykvsgltegh eyefrimaen aagisapspt spfykacdtv 22681 fkpgppgnpr vldtsrssis iawnkpiydg gseitgymve ialpeedewq ivtppaglka 22741 tsytitglte nqeykiriya mnseglgepa lvpgtpkaed rmlppeield adlrkvvtir 22801 acctlrlfvp ikgrpapevk wardhgesld kasiestssy tllivgnvnr fdsgkyiltv 22861 enssgsksaf vnvrvldtpg ppqdlkvkev tktsvtltwd pplldggski knyivekres 22921 trkaystvat nchktswkvd qlqegcsyyf rvlaeneygi glpaetaesv kaserplppg 22981 kitlmdvtrn svslswekpe hdggsrilgy ivemqtkgsd kwatcatvkv teatitgliq 23041 geeysfrvsa qnekgisdpr qlsvpviakd lvippafkll fntftvlage dlkvdvpfig 23101 rptpavtwhk dnvplkqttr vnaestenns lltikdacre dvghyvvklt nsageaietl 23161 nvivldkpgp ptgpvkmdev tadsitlswg ppkydggssi nnyivekrdt stttwqivsa 23221 tvarttikac rlktgceyqf riaaenrygk stylnseptv aqypfkvpgp pgtpvvtlss 23281 rdsmevqwne pisdggsrvi gyhlerkern silwvklnkt pipqtkfktt gleegveyef 23341 rvsaenivgi gkpskvsecy vardpcdppg rpeaiivtrn svtlqwkkpt ydggskitgy 23401 ivekkelpeg rwmkasftni idthfevtgl vedhryefrv iarnaagvfs epsestgait 23461 ardevdppri smdpkykdti vvhagesfkv dadiygkpip tiqwikgdqe lsntarleik 23521 stdfatslsv kdavrvdsgn yilkaknvag ersvtvnvkv ldrpgppegp vvisgvtaek 23581 ctlawkpplq dggsdiinyi verretsrlv wtvvdanvqt lsckvtklle gneytfrima 23641 vnkygvgepl esepvvaknp fvvpdapkap evttvtkdsm ivvwerpasd ggseilgyvl 23701 ekrdkegirw trchkrlige lrlrvtglie nhdyefrvsa enaaglseps ppsayqkacd 23761 piykpgppnn pkviditrss vflswskpiy dggceiqgyi vekcdvsvge wtmctpptgi 23821 nktnievekl lekheynfri cainkagvge hadvpgpiiv eekleapdid ldlelrkiin 23881 iraggslrlf vpikgrptpe vkwgkvdgei rdaaiidvts sftslvldnv nrydsgkytl 23941 tlenssgtks afvtvrvldt psppvnlkvt eitkdsvsit wepplldggs kiknyivekr 24001 eatrksyaav vtnchknswk idqlqegcsy yfrvtaeney giglpaqtad pikvaevpqp 24061 pgkitvddvt rnsvslswtk pehdggskii qyivemqakh sekwsecarv kslqavitnl 24121 tqgeeylfrv vavnekgrsd prslavpiva kdlviepdvk pafssysvqv gqdlkievpi 24181 sgrpkptitw tkdglplkqt trinvtdsld lttlsiketh kddggqygit vanvvgqkta 24241 sieivtldkp dppkgpvkfd dvsaesitls wnpplytggc qitnyivqkr dttttvwdvv 24301 satvarttlk vtklktgtey qfrifaenry gqsfalesdp ivaqypykep gppgtpfata 24361 iskdsmviqw hepvnnggsp vigyhlerke rnsilwtkvn ktiihdtqfk aqnleegiey 24421 efrvyaeniv gvgkasknse cyvardpcdp pgtpepimvk rneitlqwtk pvydggsmit 24481 gyivekrdlp dgrwmkasft nvietqftvs gltedqryef rviaknaaga iskpsdstgp 24541 itakdevelp rismdpkfrd tivvnagetf rleadvhgkp lptiewlrgd keieesarce 24601 ikntdfkall ivkdairidg gqyilrasnv agsksfpvnv kvldrpgppe gpvqvtgvts 24661 ekcsltwspp lqdggsdish yvvekretsr lawtvvasev vtnslkvtkl legneyvfri 24721 mavnkygvge plesapvlmk npfvlpgppk slevtniakd smtvcwnrpd sdggseiigy 24781 ivekrdrsgi rwikcnkrri tdlrlrvtgl tedheyefrv saenaagvge pspatvyyka 24841 cdpvfkpgpp tnahivdttk nsitlawgkp iydggseilg yvveickade eewqivtpqt 24901 glrvtrfeis kltehqeyki rvcalnkvgl geatsvpgtv kpedkleape ldldselrkg 24961 ivvraggsar ihipfkgrpt peitwsreeg eftdkvqiek gvnytqlsid ncdrndagky 25021 ilklenssgs ksafvtvkvl dtpgppqnla vkevrkdsaf lvweppiidg gakvknyvid 25081 krestrkaya nvsskcskts fkvenltega iyyfrvmaen efgvgvpvet vdavkaaepp 25141 sppgkvtltd vsqtsaslmw ekpehdggsr vlgyvvemqp kgtekwsiva eskvcnavvt 25201 glssgqeyqf rvkaynekgk sdprvlgvpv iakdltiqps lklpfntysi qagedlkiei 25261 pvigrprpni swvkdgeplk qttrvnveet atstvlhike gnkddfgkyt vtatnsagta 25321 tenlsvivle kpgppvgpvr fdevsadfvv isweppaytg gcqisnyive krdtttttwh 25381 mvsatvartt ikitklktgt eyqfrifaen rygksaplds kavivqypfk epgppgtpfv 25441 tsiskdqmlv qwhepvndgg tkiigyhleq keknsilwvk lnktpiqdtk fkttgldegl 25501 eyefkvsaen ivgigkpskv secfvardpc dppgrpeaiv itrnnvtlkw kkpaydggsk 25561 itgyivekkd lpdgrwmkas ftnvleteft vsglvedqry efrviarnaa gnfsepsdss 25621 gaitardeid apnasldpky kdvivvhage tfvleadirg kpipdvvwsk dgkeleetaa 25681 rmeikstiqk ttlvvkdcir tdggqyilkl snvggtksip itvkvldrpg ppegplkvtg 25741 vtaekcylaw npplqdggan ishyiiekre tsrlswtqvs tevqalnykv tkllpgneyi 25801 frvmavnkyg igeplesgpv tacnpykppg ppstpevsai tkdsmvvtwa rpvddggtei 25861 egyilekrdk egvrwtkcnk ktltdlrlrv tglteghsye frvaaenaag vgepsepsvf 25921 yracdalypp gppsnpkvtd tsrssvslaw skpiydggap vkgyvvevke aaadewttct 25981 pptglqgkqf tvtklkente ynfricains egvgepatlp gsvvaqerie ppeieldadl 26041 rkvvvlrasa tlrlfvtikg rpepevkwek aegiltdraq ievtssftml vidnvtrfds 26101 grynltlenn sgsktafvnv rvldspsapv nltirevkkd svtlsweppl idggakitny 26161 ivekrettrk ayatitnnct kttfrienlq egcsyyfrvl asneygiglp aettepvkvs 26221 epplppgrvt lvdvtrntat ikwekpesdg gskitgyvve mqtkgsekws tctqvktlea 26281 tisgltagee yvfrvaavne kgrsdprqlg vpviardiei kpsvelpfht fnvkareqlk 26341 idvpfkgrpq atvnwrkdgq tlkettrvnv sssktvtsls ikeaskedvg tyelcvsnsa 26401 gsitvpitii vldrpgppgp iridevscds itiswnppey dggcqisnyi vekkettstt 26461 whivsqavar tsikivrltt gseyqfrvca enrygkssys essavvaeyp fsppgppgtp 26521 kvvhatkstm lvtwqvpvnd ggsrvigyhl eykerssilw skankiliad tqmkvsglde 26581 glmyeyrvya eniagigkcs kscepvpard pcdppgqpev tnitrksvsl kwskphydgg 26641 akitgyiver relpdgrwlk cnytniqety fevteltedq ryefrvfarn aadsvsepse 26701 stgpiivkdd vepprvmmdv kfrdvivvka gevlkinadi agrplpvisw akdgieieer 26761 arteiistdn htlltvkdci rrdtgqyvlt lknvagtrsv avnckvldkp gppagplein 26821 gltaekcsls wgrpqedgga didyyivekr etshlawtic egelqmtsck vtkllkgney 26881 ifrvtgvnky gvgeplesva ikaldpftvp spptsleits vtkesmtlcw srpesdggse 26941 isgyiierre knslrwvrvn kkpvydlrvk stglregcey eyrvyaenaa glslpsetsp 27001 liraedpvfl psppskpkiv dsgkttitia wvkplfdgga pitgytveyk ksddtdwkts 27061 iqslrgteyt isglttgaey vfrvksvnkv gasdpsdssd pqiakereee plfdidsemr 27121 ktlivkagas ftmtvpfrgr pvpnvlwskp dtdlrtrayv dttdsrtslt ienanrndsg 27181 kytltiqnvl saasltlvvk vldtpgpptn itvqdvtkes avlswdvpen dggapvknyh 27241 iekreaskka wvsvtnncnr lsykvtnlqe gaiyyfrvsg enefgvgipa etkegvkite 27301 kpsppeklgv tsiskdsvsl twlkpehdgg srivhyvvea lekgqknwvk cavaksthhv 27361 vsglrensey ffrvfaenqa glsdprelll pvlikeqlep peidmknfps htvyvragsn 27421 lkvdipisgk plpkvtlsrd gvplkatmrf nteitaenlt inlkesvtad agryeitaan 27481 ssgttkafin ivvldrpgpp tgpvvisdit eesvtlkwep pkydggsqvt nyillkrets 27541 tavwtevsat vartmmkvmk lttgeeyqfr ikaenrfgis dhidsacvtv klpyttpgpp 27601 stpwvtnvtr esitvgwhep vsnggsavvg yhlemkdrns ilwqkanklv irtthfkvtt 27661 isagliyefr vyaenaagvg kpshpsepvl aidacepprn vritdiskns vslswqqpaf 27721 dggskitgyi verrdlpdgr wtkasftnvt etqfiisglt qnsqyefrvf arnavgsisn 27781 psevvgpitc idsyggpvid lpleytevvk yragtsvklr agisgkpapt iewykddkel 27841 qtnalvcven ttdlasilik dadrlnsgcy elklrnamgs asatirvqil dkpgppggpi 27901 efktvtaeki tllwrppadd ggakithyiv ekretsrvvw smvsehleec iitttkiikg 27961 neyifrvrav nkygigeple sdsvvaknaf vtpgppgipe vtkitknsmt vvwsrpiadg 28021 gsdisgyfle krdkkslgwf kvlketirdt rqkvtglten sdyqyrvcav naagqgpfse 28081 psefykaadp idppgppaki riadstkssi tlgwskpvyd ggsavtgyvv eirqgeeeew 28141 ttvstkgevr tteyvvsnlk pgvnyyfrvs avncagqgep iemnepvqak dileapeidl 28201 dvalrtsvia kagedvqvli pfkgrppptv twrkdeknlg sdarysient dssslltipq 28261 vtrndtgkyi ltiengvgep ksstvsvkvl dtpaacqklq vkhvsrgtvt llwdpplidg 28321 gspiinyvie krdatkrtws vvshkcssts fklidlsekt pfffrvlaen eigigepcet 28381 tepvkaaevp apirdlsmkd stktsvilsw tkpdfdggsv iteyvverkg kgeqtwshag 28441 isktceievs qlkeqsvlef rvfaknekgl sdpvtigpit vkeliitpev dlsdipgaqv 28501 tvrighnvhl elpykgkpkp siswlkdglp lkesefvrfs ktenkitlsi knakkehggk 28561 ytvildnavc riavpitvit lgppskpkgp irfdeikads vilswdvped ngggeitcys 28621 iekretsqtn wkmvcssvar ttfkvpnlvk daeyqfrvra enrygvsqpl vssiivakhq 28681 fripgppgkp viynvtsdgm sltwdapvyd ggsevtgfhv ekkernsilw qkvntspisg 28741 reyratglve gldyqfrvya ensaglssps dpskftlavs pvdppgtpdy idvtretitl 28801 kwnpplrdgg skivgysiek rqgnerwvrc nftdvsecqy tvtglspgdr yefriiarna 28861 vgtisppsqs sgiimtrden vppivefgpe yfdgliiksg eslrikalvq grpvprvtwf 28921 kdgveiekrm nmeitdvlgs tslfvrdatr dhrgvytvea knasgsakae ikvkvqdtpg 28981 kvvgpirftn itgekmtlww daplndgcap ithyiiekre tsrlawalie dkceaqsyta 29041 iklingneyq frvsavnkfg vgrpldsdpv vaqiqytvpd apgipepsni tgnsitltwa 29101 rpesdggsei qqyilerrek kstrwvkvis krpisetrfk vtgltegney efhvmaenaa 29161 gvgpasgisr likcrepvnp pgpptvvkvt dtskttvsle wskpvfdggm eiigyiiemc 29221 kadlgdwhkv naeacvktry tvtdlqagee ykfrvsaing agkgdscevt gtikavdrlt 29281 apeldidanf kqthvvraga sirlfiayqg rptptavwsk pdsnlslrad ihttdsfstl 29341 tvencnrnda gkytltvenn sgsksitftv kvldtpgppg pitfkdvtrg satlmwdapl 29401 ldggarihhy vvekreasrr swqvisekct rqifkvndla egvpyyfrvs avneygvgep 29461 yempepivat eqpapprrld vvdtskssav lawlkpdhdg gsritgylle mrqkgsdfwv 29521 eaghtkqltf tverlvekte yefrvkaknd agysepreaf ssviikepqi eptadltgit 29581 nqlitckags pftidvpisg rpapkvtwkl eemrlketdr vsitttkdrt tltvkdsmrg 29641 dsgryfltle ntagvktfsv tvvvigrpgp vtgpievssv saescvlswg epkdgggtei 29701 tnyivekres gttawqlvns svkrtqikvt hltkymeysf rvssenrfgv skplesapii 29761 aehpfvppsa ptrpevyhvs anamsirwee pyhdggskii gywvekkern tilwvkenkv 29821 pclecnykvt glvegleyqf rtyalnaagv skaseasrpi maqnpvdapg rpevtdvtrs 29881 tvsliwsapa ydggskvvgy iierkpvsev gdgrwlkcny tivsdnfftv talsegdtye 29941 frvlaknaag viskgsestg pvtcrdeyap pkaeldarlh gdlvtirags dlvldaavgg 30001 kpepkiiwtk gdkeldlcek vslqytgkra tavikfcdrs dsgkytltvk nasgtkavsv 30061 mvkvldspgp cgkltvsrvt qekctlawsl pqedggaeit hyiverrets rlnwvivege 30121 cptlsyvvtr liknneyifr vravnkygpg vpvesepiva rnsftipspp gipeevgtgk 30181 ehiiiqwtkp esdggneisn ylvdkrekks lrwtrvnkdy vvydtrlkvt slmegcdyqf 30241 rvtavnaagn sepseasnfi screpsytpg ppsaprvvdt tkhsislawt kpmydggtdi 30301 vgyvlemqek dtdqwyrvht natirnteft vpdlkmgqky sfrvaavnvk gmseysesia 30361 eiepveriei pdleladdlk ktvtiragas lrlmvsvsgr pppvitwskq gidlasraii 30421 dttesyslli vdkvnrydag kytieaenqs gkksatvlvk vydtpgpcps vkvkevsrds 30481 vtitweipti dggapvnnyi vekreaamra fktvttkcsk tlyrisglve gtmyyfrvlp 30541 eniygigepc etsdavlvse vplvpaklev vdvtkstvtl awekplydgg srltgyvlea 30601 ckagterwmk vvtlkptvle htvtslnege qylfriraqn ekgvsepret vtavtvqdlr 30661 vlptidlstm pqktihvpag rpvelvipia grpppaaswf fagsklrese rvtvethtkv 30721 akltiretti rdtgeytlel knvtgttset ikviildkpg pptgpikide idatsitisw 30781 eppeldggap lsgyvveqrd ahrpgwlpvs esvtrstfkf trltegneyv frvaatnrfg 30841 igsylqsevi ecrssiripg ppetlqifdv srdgmtltwy ppeddggsqv tgyiverkev 30901 radrwvrvnk vpvtmtryrs tgltegleye hrvtainarg sgkpsrpskp ivamdpiapp 30961 gkpqnprvtd ttrtsvslaw svpedeggsk vtgyliemqk vdqhewtkcn ttptkireyt 31021 lthlpqgaey rfrvlacnag gpgepaevpg tvkvtemley pdyelderyq egifvrqggv 31081 irltipikgk pfpickwtke gqdiskrami atsethtelv ikeadrgdsg tydlvlenkc 31141 gkkavyikvr vigspnspeg pleyddiqvr svrvswrppa ddggadilgy ilerrevpka 31201 awytidsrvr gtslvvkglk enveyhfrvs aenqfgiskp lkseepvtpk tplnppepps 31261 nppevldvtk ssvslswsrp kddggsrvtg yyierketst dkwvrhnktq itttmytvtg 31321 lvpdaeyqfr iiaqndvgls etspasepvv ckdpfdkpsq pgeleilsis kdsvtlqwek 31381 pecdggkeil gywveyrqsg dsawkksnke rikdkqftig glleateyef rvfaenetgl 31441 srprrtamsi ktkltsgeap girkemkdvt tklgeaaqls cqivgrplpd ikwyrfgkel 31501 iqsrkykmss dgrthtltvm teeqedegvy tciatnevge vetssklllq atpqfhpgyp 31561 lkekyygavg stlrlhvmyi grpvpamtwf hgqkllqnse nitientehy thlvmknvqr 31621 kthagkykvq lsnvfgtvda ildveiqdkp dkptgpivie allknsavis wkppaddggs 31681 witnyvvekc eakegaewql vssaisvttc rivnltenag yyfrvsaqnt fgisdplevs 31741 svviikspfe kpgapgkpti tavtkdscvv awkppasdgg akirnyylek rekkqnkwis 31801 vtteeiretv fsvknliegl eyefrvkcen lggesewsei sepitpksdv piqaphfkee 31861 lrnlnvryqs natlvckvtg hpkpivkwyr qgkeiiadgl kyriqefkgg yhqliiasvt 31921 dddatvyqvr atnqggsvsg taslevevpa kihlpktleg mgavhalrge vvsikipfsg 31981 kpdpvitwqk gqdlidnngh yqvivtrsft slvfpngver kdagfyvvca knrfgidqkt 32041 veldvadvpd pprgvkvsdv srdsvnltwt epasdggski tnyivekcat taerwlrvgq 32101 aretrytvin lfgktsyqfr viaenkfgls kpsepsepti tkedktramn ydeevdetre 32161 vsmtkashss tkelyekymi aedlgrgefg ivhrcvetss kktymakfvk vkgtdqvlvk 32221 keisilniar hrnilhlhes fesmeelvmi fefisgldif erintsafel nereivsyvh 32281 qvcealqflh shnighfdir peniiyqtrr sstikiiefg qarqlkpgdn frllftapey 32341 yapevhqhdv vstatdmwsl gtlvyvllsg inpflaetnq qiienimnae ytfdeeafke 32401 isieamdfvd rllvkerksr mtasealqhp wlkqkiervs tkvirtlkhr ryyhtlikkd 32461 lnmvvsaari scggairsqk gvsvakvkva sieigpvsgq imhavgeegg hvkyvckien 32521 ydqstqvtwy fgvrqlense kyeityedgv ailyvkditk lddgtyrckv vndygedssy 32581 aelfvkgvre vydyycrrtm kkikrrtdtm rllerppeft lplynktayv genvrfgvti 32641 tvhpephvtw yksgqkikpg dndkkytfes dkglyqltin svttdddaey tvvarnkyge 32701 dsckakltvt lhppptdstl rpmfkrllan aecqegqsvc feirvsgipp ptlkwekdgq 32761 plslgpniei ihegldyyal hirdtlpedt gyyrvtatnt agstscqahl qverlrykkq 32821 efkskeeher hvqkqidktl rmaeilsgte svpltqvake alreaavlyk pavstktvkg 32881 efrleieekk eerklrmpyd vpeprkykqt tieedqrikq fvpmsdmkwy kkirdqyemp 32941 gkldrvvqkr pkrirlsrwe qfyvmplpri tdqyrpkwri pklsqddlei vrparrrtps 33001 pdydfyyrpr rrslgdisde elllpiddyl amkrteeerl rleeelelgf sasppsrspp 33061 hfelsslrys spqahvkvee trkdfrysty hiptkaeast syaelrerha qaayrqpkqr 33121 qrimaerede ellrpvtttq hlseykseld fmskeeksrk ksrrqrevte iteieeeyei 33181 skhaqresss sasrllrrrr slsptyielm rpvselirsr pqpaeeyedd terrsptper 33241 trprspspvs serslsrfer sarfdifsry esmkaalktq ktserkyevl sqqpftldha 33301 pritlrmrsh rvpcgqntrf ilnvqskpta evkwyhngve lqesskihyt ntsgvltlei 33361 ldchtddsgt yravctnykg easdyatldv tggdyttyas qrrdeevprs vfpeltrtea 33421 yavssfkkts emeasssvre vksqmtetre slssyehsas aemksaalee ksleeksttr 33481 kikttlaari ltkprsmtvy egesarfscd tdgepvptvt wlrkgqvlst sarhqvtttk 33541 ykstfeissv qasdegnysv vvensegkqe aeftltiqka rvtekavtsp prvkspeprv 33601 kspeavkspk rvkspepshp kavsptetkp tptekvqhlp vsappkitqf lkaeaskeia 33661 kltcvvessv lrakevtwyk dgkklkengh fqfhysadgt yelkinnlte sdqgeyvcei 33721 sgeggtsktn lqfmgqafks ihekvskise tkksdqktte stvtrktepk apepisskpv 33781 ivtglqdttv ssdsvakfav katgeprpta iwtkdgkait qggkyklsed kggffleihk 33841 tdtsdsglyt ctvknsagsv sssckltika ikdteaqkvs tqktseitpq kkavvqeeis 33901 qkalrseeik mseaksqekl alkeeaskvl iseevkksaa tsleksivhe eitktsqase 33961 evrthaeika fstqmsineg qrlvlkania gatdvkwvln gveltnseey rygvsgsdqt 34021 ltikqashrd egiltciskt kegivkcqyd ltlskelsda pafisqprsq ninegqnvlf 34081 tceisgepsp eiewfknnlp isissnvsis rsrnvyslei rnasvsdsgk ytikaknfrg 34141 qcsataslmv lplveepsre vvlrtsgdts lqgsfssqsv qmsaskqeas fssfssssas 34201 smtemkfasm saqsmssmqe sfvemssssf mgisnmtqle sstskmlkag irgippkiea 34261 lpsdisideg kvltvacaft geptpevtws cggrkihsqe qgrfhientd dlttliimdv 34321 qkqdgglytl slgnefgsds atvnihirsi';

} else if (randomizer <= 100) {

peptidesequence = '1 mdqpqfsgap rfltrpkafv vsvgkdatls cqivgnptpq vswekdqqpv aagarfrlaq 61 dgdlyrltil dlalgdsgqy vcrarnaige afaavglqvd aeaacaeqap hfllrptsir 121 vregseatfr crvggsprpa vswskdgrrl gepdgprvrv eelgeasalr iraarprdgg 181 tyevraenpl gaasaaaalv vdsdaadtas rpgtstaall ahlqrrream raegapaspp 241 stgtrtctvt egkharlscy vtgepkpetv wkkdgqlvte grrhvvyeda qenfvlkilf 301 ckqsdrglyt ctasnlvgqt yssvlvvvre pavpfkkrlq dlevrekesa tflcevpqps 361 teaawfkeet rlwasakygi eeegterrlt vrnvsaddda vyicetpegs rtvaelavqg 421 nllrklprkt avrvgdtamf cvelavpvgp vhwlrnqeev vaggrvaisa egtrhtltis 481 qccledvgqv afmagdcqts tqfcvsaprk pplqppvdpv vkarmessvi lswsppphge 541 rpvtidgylv ekkklgtytw ircheaewva tpeltvadva eegnfqfrvs alnsfgqspy 601 lefpgtvhla pklavrtplk avqaveggev tfsvdltvas agewfldgqa lkassvyeih 661 cdrtrhtlti revpaslhga qlkfvangie ssirmevraa pgltankppa aaarevlarl 721 heeaqllael sdqaaavtwl kdgrtlspgp kyevqasagr rvllvrdvar ddaglyecvs 781 rggriayqls vqglarflhk dmagscvdav aggpaqfece tseahvhvhw ykdgmelghs 841 gerflqedvg trhrlvaatv trqdegtysc rvgedsvdfr lrvsepkvvf akeqlarrkl 901 qaeagasatl scevaqaqte vtwykdgkkl sssskvcmea tgctrrlvvq qagqadagey 961 sceaggqrls fhldvkepkv vfakdqvahs evqaeagasa tlscevaqaq tevmwykdgk 1021 klssslkvhv eakgcrrrlv vqqagktdag dysceargqr vsfrlhitep kmmfakeqsv 1081 hnevqaeaga samlscevaq aqtevtwykd gkklsssskv gmevkgctrr lvlpqagkad 1141 ageysceagg qrvsfhlhit epkgvfakeq svhnevqaea gttamlscev aqpqtevtwy 1201 kdgkklssss kvrmevkgct rrlvvqqvgk adageyscea ggqrvsfqlh itepkavfak 1261 eqlvhnevrt eagasatlsc evaqaqtevt wykdgkklss sskvrieaag cmrqlvvqqa 1321 gqadageytc eaggqrlsfh ldvsepkavf akeqlahrkv qaeagaiatl scevaqaqte 1381 vtwykdgkkl sssskvrmea vgctrrlvvq qacqadtgey sceaggqrls fsldvaepkv 1441 vfakeqpvhr evqaqagast tlscevaqaq tevmwykdgk klsfsskvrm eavgctrrlv 1501 vqqagqavag eysceagsqr lsfhlhvaep kavfakeqpa srevqaeagt satlscevaq 1561 aqtevtwykd gkklsssskv rmeavgctrr lvvqeagqad ageysckagd qrlsfhlhva 1621 epkvvfakeq pahrevqaea gasatlscev aqaqtevtwy kdgkklssss kvrveavgct 1681 rrlvvqqagq aeageyscea ggqqlsfrlq vaelepqise rpcrreplvv kehediilta 1741 tlatpsaatv twlkdgveir rskrhetasq gdthtltvhg aqvldsaiys crvgaegqdf 1801 pvqveevaak fcrllepvcg elggtvtlac elspacaevv wrcgntqlrv gkrfqmvaeg 1861 pvrsltvlgl raedageyvc esrddhtsaq ltvsvprvvk fmsglstvva eeggeatfqc 1921 vvspsdvavv wfrdgallqp sekfaisqsg ashsltisdl vledagqitv eaegasssaa 1981 lrvreapvlf kkklepqtve erssvtleve ltrpwpelrw trnatalapg knveihaega 2041 rhrlvlhnvg fadrgffgce tpddktqakl tvemrqvrlv rglqaveare qgtatmevql 2101 shadvdgswt rdglrfqqgp tchlavrgpm htltlsglrp edsglmvfka egvhtsarlv 2161 vtelpvsfsr plqdvvttek ekvtlecels rpnvdvrwlk dgvelragkt maiaaqgacr 2221 sltiyrcefa dqgvyvcdah daqssasvkv qgrtytliyr rvlaedagei qfvaenaesr 2281 aqlrvkelpv tlvrplrdki amekhrgvle cqvsrasaqv rwfkgsqelq pgpkyelvsd 2341 glyrkliisd vhaededtyt cdagdvktsa qffveeqsit ivrglqdvtv mepapawfec 2401 etsipsvrpp kwllgktvlq aggnvgleqe gtvhrlmlrr tcstmtgpvh ftvgksrssa 2461 rlvvsdipvv ltrplepktg relqsvvlsc dfrpapkavq wykddtplsp sekfkmsleg 2521 qmaelrilrl mpadagvyrc qagsahsste vtvearevtv tgplqdaeat eegwasfsce 2581 lshedeevew slngmplynd sfheishkgr rhtlvlksiq radagivras slkvstsarl 2641 evrvkpvvfl kalddlsaee rgtlalqcev sdpeahvvwr kdgvqlgpsd kydflhtagt 2701 rglvvhdvsp edaglytchv gseetrarvr vhdlhvgitk rlktmevleg escsfecvls 2761 hesasdpamw tvggktvgss srfqatrqgr kyilvvreaa psdagevvfs vrgltskasl 2821 ivrerpaaii kpledqwvap gedvelrcel sragtpvhwl kdrkairksq kydvvcegtm 2881 amlvirgasl kdageytcev easkstaslh veekancfte eltnlqveek gtavftckte 2941 hpaatvtwrk gllelrasgk hqpsqegltl rltisaleka dsdtytcdig qaqsraqllv 3001 qgrrvhiied ledvdvqegs satfrcrisp anyepvhwfl dktplhanel neidaqpggy 3061 hvltlrqlal kdsgtiyfea gdqrasaalr vtekpsvfsr eltdatiteg edltlvcets 3121 tcdipvcwtk dgktlrgsar cqlsheghra qllitgatlq dsgrykceag gacsssivrv 3181 harpvrfqea lkdlevlegg aatlrcvlss vaapvkwcyg nnvlrpgdky slrqegamle 3241 lvvrnlrpqd sgryscsfgd qttsatltvt alpaqfigkl rnkeategat atlrcelska 3301 apvewrkgse tlrdgdrycl rqdgamcelq irglamvdaa eyscvcgeer tsasltirpm 3361 pahfigrlrh qesiegatat lrcelskaap vewrkgresl rdgdrhslrq dgavcelqic 3421 glavadagey scvcgeerts atltvkalpa kfteglrnee avegatamlw celskvapve 3481 wrkgpenlrd gdryilrqeg trcelqicgl amadageylc vcgqertsat ltiralparf 3541 iedvknqear egatavlqce lnsaapvewr kgsetlrdgd ryslrqdgtk celqirglam 3601 adtgeyscvc gqertsamlt vralpikfte glrneeateg atavlrcels kmapvewwkg 3661 hetlrdgdrh slrqdgarce lqirglvaed ageylcmcgk ertsamltvr ampskfiegl 3721 rneeategdt atlwcelska apvewrkghe tlrdgdrhsl rqdgsrcelq irglavvdag 3781 eyscvcgqer tsatltvral parfiedvkn qearegatav lqcelskaap vewrkgsetl 3841 rggdryslrq dgtrcelqih glsvadtgey scvcgqerts atltvrapqp vfreplqslq 3901 aeegstatlq celseptatv vwskgglqlq angrreprlq gctaelvlqd lqredtgeyt 3961 ctcgsqatsa tltvtaapvr flrelqhqev deggtahlcc elsragasve wrkgslqlfp 4021 cakyqmvqdg aaaellvrgv eqedagdytc dtghtqsmas lsvrvprpkf ktrlqsleqe 4081 tgdiarlccq lsdaesgavv qwlkegvelh agpkyemrsq gatrellihq leakdtgeya 4141 cvtggqktaa slrvtepevt ivrglvdaev tadedvefsc evsragatgv qwclqglplq 4201 snevtevavr dgrihtlrlk gvtpedagtv sfhlgnhass aqltvrapev tileplqdvq 4261 lsegqdasfq crlsrasgqe arwalggvpl qanemnditv eqgtlhlltl hkvtledagt 4321 vsfhvgtcss eaqlkvtakn tvvrglenve aleggealfe cqlsqpevaa htwllddepv 4381 htsenaevvf fenglrhlll lknlrpqdsc rvtflagdmv tsafltvrgw rleileplkn 4441 aavragaqac ftctlseavp vgeaswying aavqpddsdw tvtadgshha lllrsaqphh 4501 agevtfacrd avasarltvl glpdppedae vvarsshtvt lswaapmsdg ggglcgyrve 4561 vkegatgqwr lchelvpgpe cvvdglapge tyrfrvaavg pvgagepvhl pqtvrlaepp 4621 kpvppqpsap esrqvaaged vslelevvae ageviwhkgm eriqpggrfe vvsqgrqqml 4681 vikgftaedq geyhcglaqg sicpaaatfq valspasvde apqpslppea aqegdlhllw 4741 ealarkrrms reptldsise lpeedgrsqr lpqeaeevap dlsegystad elartgdadl 4801 shtssddesr agtpslvtyl kkagrpgtsp laskvgapaa psvkpqqqqe plaavrpplg 4861 dlstkdlgdp smdkaavkiq aafkgykvrk emkqqegpmf shtfgdteaq vgdalrlecv 4921 vaskadvrar wlkdgveltd grhhhidqlg dgtcsllitg ldradagcyt cqvsnkfgqv 4981 thsacvvvsg seseaesssg gelddafrra arrlhrlfrt kspaevsdee lflsadegpa 5041 epeepadwqt yredehfici rfealtearq avtrfqemfa tlgigveikl veqgprrvem 5101 cisketpapv vppeplpsll tsdaapvflt elqnqevqdg ypvsfdcvvt gqpmpsvrwf 5161 kdgklleedd hyminedqqg ghqliitavv padmgvyrcl aensmgvsst kaelrvdlts 5221 tdydtaadat esssyfsaqg ylssreqegt esttdegqlp qvveelrdlq vapgtrlakf 5281 qlkvkgypap rlywfkdgqp ltasahirmt dkkilhtlei isvtredsgq yaayisnamg 5341 aayssarllv rgpdepeekp asdvheqlvp prmlerftpk kvkkgssitf svkvegrpvp 5401 tvhwlreeae rgvlwigpdt pgytvassaq qhslvlldvg rqhqgtytci asnaagqalc 5461 saslhvsglp kveeqekvke alistflqgt tqaisaqgle tasfadlggq rkeeplaake 5521 alghlslaev gteeflqklt sqitemvsak itqaklqvpg gdsdedsktp sasprhgrsr 5581 psssiqesss esedgdarge ifdiyvvtad ylplgaeqda itlregqyve vldaahplrw 5641 lvrtkptkss psrqgwvspa yldrrlklsp ewgaaeapef pgeavsedey karlssviqe 5701 llsseqafve elqflqshhl qhlercphvp iavagqkavi frnvrdigrf hssflqelqq 5761 cdtdddvamc fiknqaafeq yleflvgrvq aesvvvstai qefykkyaee allagdpsqp 5821 pppplqhyle qpvervqryq allkelirnk arnrqncall eqayavvsal pqraenklhv 5881 slmenypgtl qalgepirqg hfivwegapg armpwkghnr hvflfrnhlv ickprrdsrt 5941 dtvsyvfrnm mklssidlnd qvegddrafe vwqeredsvr kyllqartai iksswvkeic 6001 giqqrlalpv wrppdfeeel adctaelget vklacrvtgt pkpviswykd gkavqvdphh 6061 iliedpdgsc alildsltgv dsgqymcfaa saagncstlg kilvqvpprf vnkvraspfv 6121 egedaqftct iegapypqir wykdgalltt gnkfqtlsep rsgllvlvir aaskedlgly 6181 ecelvnrlgs arasaelriq spmlqaqeqc hreqlvaave vteqetkvpk ktviieetit 6241 tvvksprgqr rspskspsrs psrcsasplr pgllapdlly lpgagqprrp eaepgqkpvv 6301 ptlyvteaea hspalpglsg pqpkwvevee tievrvkkmg pqgvspttev prsssghlft 6361 lpgatpggdp nsnnsnnkll aqeawaqgta mvgvreplvf rvdargsvdw aasgmgslee 6421 egtmeeagee egedgdafvt eesqdthslg drdpkilthn grmltladle dyvpgegetf 6481 hcggpgpgap ddppcevsvi qreigeptvg qpvllsvgha lgprgplglf rpeprgaspp 6541 gpqvrslegt sfllreapar pvgsapwtqs fctrirrsad sgqssfttel stqtvnfgtv 6601 getvtlhicp drdgdeaaqp';

} else if (randomizer <= 150) {

peptidesequence = ' 1 mlwweevedc yeredvqkkt ftkwvnaqfs kfgkqhienl fsdlqdgrrl ldllegltgq 61 klpkekgstr vhalnnvnka lrvlqnnnvd lvnigstdiv dgnhkltlgl iwniilhwqv 121 knvmknimag lqqtnsekil lswvrqstrn ypqvnvinft tswsdglaln alihshrpdl 181 fdwnsvvcqq satqrlehaf niaryqlgie klldpedvdt typdkksilm yitslfqvlp 241 qqvsieaiqe vemlprppkv tkeehfqlhh qmhysqqitv slaqgyerts spkprfksya 301 ytqaayvtts dptrspfpsq hleapedksf gsslmesevn ldryqtalee vlswllsaed 361 tlqaqgeisn dvevvkdqfh thegymmdlt ahqgrvgnil qlgskligtg klsedeetev 421 qeqmnllnsr weclrvasme kqsnlhrvlm dlqnqklkel ndwltkteer trkmeeeplg 481 pdledlkrqv qqhkvlqedl eqeqvrvnsl thmvvvvdes sgdhataale eqlkvlgdrw 541 anicrwtedr wvllqdillk wqrlteeqcl fsawlseked avnkihttgf kdqnemlssl 601 qklavlkadl ekkkqsmgkl yslkqdllst lknksvtqkt eawldnfarc wdnlvqklek 661 staqisqavt ttqpsltqtt vmetvttvtt reqilvkhaq eelpppppqk krqitvdsei 721 rkrldvdite lhswitrsea vlqspefaif rkegnfsdlk ekvnaierek aekfrklqda 781 srsaqalveq mvnegvnads ikqaseqlns rwiefcqlls erlnwleyqn niiafynqlq 841 qleqmtttae nwlkiqpttp septaiksql kickdevnrl sdlqpqierl kiqsialkek 901 gqgpmfldad fvaftnhfkq vfsdvqarek elqtifdtlp pmryqetmsa irtwvqqset 961 klsipqlsvt dyeimeqrlg elqalqsslq eqqsglyyls ttvkemskka pseisrkyqs 1021 efeeiegrwk klssqlvehc qkleeqmnkl rkiqnhiqtl kkwmaevdvf lkeewpalgd 1081 seilkkqlkq crllvsdiqt iqpslnsvne ggqkikneae pefasrlete lkelntqwdh 1141 mcqqvyarke alkgglektv slqkdlsemh ewmtqaeeey lerdfeyktp delqkaveem 1201 krakeeaqqk eakvklltes vnsviaqapp vaqealkkel etlttnyqwl ctrlngkckt 1261 leevwacwhe llsylekank wlnevefklk ttenipggae eisevldsle nlmrhsednp 1321 nqirilaqtl tdggvmdeli neeletfnsr wrelheeavr rqklleqsiq saqetekslh 1381 liqesltfid kqlaayiadk vdaaqmpqea qkiqsdltsh eisleemkkh nqgkeaaqrv 1441 lsqidvaqkk lqdvsmkfrl fqkpanfeqr lqeskmilde vkmhlpalet ksveqevvqs 1501 qlnhcvnlyk slsevkseve mviktgrqiv qkkqtenpke ldervtalkl hynelgakvt 1561 erkqqlekcl klsrkmrkem nvltewlaat dmeltkrsav egmpsnldse vawgkatqke 1621 iekqkvhlks itevgealkt vlgkketlve dklsllnsnw iavtsraeew lnllleyqkh 1681 metfdqnvdh itkwiiqadt lldesekkkp qqkedvlkrl kaelndirpk vdstrdqaan 1741 lmanrgdhcr klvepqisel nhrfaaishr iktgkasipl keleqfnsdi qkllepleae 1801 iqqgvnlkee dfnkdmnedn egtvkellqr gdnlqqritd erkreeikik qqllqtkhna 1861 lkdlrsqrrk kaleishqwy qykrqaddll kclddiekkl aslpeprder kikeidrelq 1921 kkkeelnavr rqaeglsedg aamaveptqi qlskrwreie skfaqfrrln faqihtvree 1981 tmmvmtedmp leisyvpsty lteithvsqa lleveqllna pdlcakdfed lfkqeeslkn 2041 ikdslqqssg ridiihskkt aalqsatpve rvklqealsq ldfqwekvnk mykdrqgrfd 2101 rsvekwrrfh ydikifnqwl teaeqflrkt qipenwehak ykwylkelqd gigqrqtvvr 2161 tlnatgeeii qqssktdasi lqeklgslnl rwqevckqls drkkrleeqk nilsefqrdl 2221 nefvlwleea dniasiplep gkeqqlkekl eqvkllveel plrqgilkql netggpvlvs 2281 apispeeqdk lenklkqtnl qwikvsralp ekqgeieaqi kdlgqlekkl edleeqlnhl 2341 llwlspirnq leiynqpnqe gpfdvketei avqakqpdve eilskgqhly kekpatqpvk 2401 rkledlssew kavnrllqel rakqpdlapg lttigasptq tvtlvtqpvv tketaiskle 2461 mpsslmlevp aladfnrawt eltdwlslld qviksqrvmv gdledinemi ikqkatmqdl 2521 eqrrpqleel itaaqnlknk tsnqeartii tdrieriqnq wdevqehlqn rrqqlnemlk 2581 dstqwleake eaeqvlgqar akleswkegp ytvdaiqkki tetkqlakdl rqwqtnvdva 2641 ndlalkllrd ysaddtrkvh miteninasw rsihkrvser eaaleethrl lqqfpldlek 2701 flawlteaet tanvlqdatr kerlledskg vkelmkqwqd lqgeieahtd vyhnldensq 2761 kilrslegsd davllqrrld nmnfkwselr kkslnirshl eassdqwkrl hlslqellvw 2821 lqlkddelsr qapiggdfpa vqkqndvhra fkrelktkep vimstletvr iflteqpleg 2881 leklyqepre lppeeraqnv trllrkqaee vnteweklnl hsadwqrkid etlerlrelq 2941 eatdeldlkl rqaevikgsw qpvgdllids lqdhlekvka lrgeiaplke nvshvndlar 3001 qlttlgiqls pynlstledl ntrwkllqva vedrvrqlhe ahrdfgpasq hflstsvqgp 3061 weraispnkv pyyinhetqt tcwdhpkmte lyqsladlnn vrfsayrtam klrrlqkalc 3121 ldllslsaac daldqhnlkq ndqpmdilqi inclttiydr leqehnnlvn vplcvdmcln 3181 wllnvydtgr tgrirvlsfk tgiislckah ledkyrylfk qvasstgfcd qrrlglllhd 3241 siqiprqlge vasfggsnie psvrscfqfa nnkpeieaal fldwmrlepq smvwlpvlhr 3301 vaaaetakhq akcnickecp iigfryrslk hfnydicqsc ffsgrvakgh kmhypmveyc 3361 tpttsgedvr dfakvlknkf rtkryfakhp rmgylpvqtv legdnmetpv tlinfwpvds 3421 apasspqlsh ddthsriehy asrlaemens ngsylndsis pnesiddehl liqhycqsln 3481 qdsplsqprs paqilisles eergeleril adleeenrnl qaeydrlkqq hehkglsplp 3541 sppemmptsp qsprdaelia eakllrqhkg rlearmqile dhnkqlesql hrlrqlleqp 3601 qaeakvngtt vsspstslqr sdssqpmllr vvgsqtsdsm geedllsppq dtstgleevm 3661 eqlnnsfpss rgrntpgkpm redtm';

} else {

peptidesequence = '1 maddedyeev veyyteevvy eevpgetitk iyettttrts dyeqsetskp alaqpalaqp 61 asakpverrk virkkvdpsk fmtpyiahsq kmqdlfspnk ykekfektkg qpyasttdtp 121 elrrikkvqd qlsevkyrmd gdvaktichv dekakdieha kkvsqqvskv lykqnwedtk 181 dkyllppdap elvqavknta mfskklyted weadkslfyp yndspelrrv aqaqkalsdv 241 aykkglaeqq aqftpladpp diefakkvtn qvskqkyked yenkikgkws etpcfevana 301 rmnadnistr kyqedfenmk dqiyfmqtet peykmnkkag vaaskvkyke dyeknkgkad 361 ynvlpasenp qlrqlkaagd alsdklyken yektkaksin ycetpkfkld tvlqnfssdk 421 kykdsylkdi lghyvgsfed pyhshcmkvt aqnsdknyka eyeedrgkgf fpqtitqeye 481 aikkldqckd htykvhpdkt kftqvtdspv llqaqvnskq lsdlnykakh esekfkchip 541 pdtpafiqhk vnaynlsdnl ykqdwekska kkfdikvdai pllaakantk ntsdvmykkd 601 yeknkgkmig vlsinddpkm lhslkvaknq sdrlykenye ktkaksmnyc etpkyqldtq 661 lknfsearyk dlyvkdvlgh yvgsmedpyh thcmkvaaqn sdksykaeye edkgkcyfpq 721 titqeyeaik kldqckdhty kvhpdktkft avtdspvllq aqlntkqlsd lnykakhege 781 kfkchipada pqfiqhrvna ynlsdnvykq dwekskakkf dikvdaipll aakantknts 841 dvmykkdyek skgkmigals inddpkmlhs lktaknqsdr eyrkdyeksk tiytapldml 901 qvtqakksqa iasdvdykhi lhsysyppds invdlakkay alqsdveyka dynswmkgcg 961 wvpfgsleme kakrasdiln ekkyrqhpdt lkftsiedap itvqskinqa qrsdiaykak 1021 geeiihkynl ppdlpqfiqa kvnaynisen mykadlkdls kkgydlrtda ipiraakaar 1081 qaasdvqykk dyekakgkmv gfqslqddpk lvhymnvaki qsdreykkdy ektkskyntp 1141 hdmfnvvaak kaqdvvsnvn ykhslhhyty lpdamdlels knmmqiqsdn vykedynnwm 1201 kgigwipigs ldvekvkkag dalnekkyrq hpdtlkftsi vdspvmvqak qntkqvsdil 1261 ykakgedvkh kytmspdlpq flqakcnayn isdvcykrdw ydliakgnnv lgdaipitaa 1321 kasrniasdy kykeayeksk gkhvgfrslq ddpklvhymn vaklqsdrey kknyentkts 1381 yhtpgdmvsi taakmaqdva tnvnykqplh hytylpdams lehtrnvnqi qsdnvykdey 1441 nsflkgigwi pigslevekv kkagdalner kyrqhpdtvk ftsvpdsmgm vlaqhntkql 1501 sdlnykvege klkhkytidp elpqfiqakv nalnmsdahy kadwkktiak gydlrpdaip 1561 ivaakssrni asdckykeay ekakgkqvgf lslqddpklv hymnvakiqs dreykkgyea 1621 sktkyhtpld mvsvtaakks qevatnanyr qsyhhytllp dalnvehsrn amqiqsdnly 1681 ksdftnwmkg igwvpiesle vekakkagei lsekkyrqhp eklkftyamd tmeqalnksn 1741 klnmdkrlyt ekwnkdktti hvmpdtpdil lsrvnqitms dklykagwee ekkkgydlrp 1801 daiaikaara srdiasdyky kkayeqakgk higfrsledd pklvhfmqva kmqsdreykk 1861 gyeksktsfh tpvdmlsvva akksqevatn anyrnvihty nmlpdamsfe laknmmqiqs 1921 dnqykadyad fmkgigwlpl gsleaeknkk ameiisekky rqhpdtlkys tlmdsmnmvl 1981 aqnnakimne hlykqawead ktkvhimpdi pqiilakana inmsdklykl sleeskkkgy 2041 dlrpdaipik aakasrdias dykykynyek gkgkmvgfrs leddpklvhs mqvakmqsdr 2101 eykknyentk tsyhtpadml svtaakdaqa nitntnykhl ihkyillpda mnieltrnmn 2161 riqsdneykq dynewykglg wspagsleve kakkateyas dqkyrqhpsn fqfkkltdsm 2221 dmvlakqnah tmnkhlytid wnkdktkihv mpdtpdilqa kqnqtlysqk lyklgweeal 2281 kkgydlpvda isvqlakasr diasdykykq gyrkqlghhv gfrslqddpk lvlsmnvakm 2341 qsereykkdf ekwktkfssp vdmlgvvlak kcqelvsdvd yknylhqwtc lpdqndvvqa 2401 kkvyelqsen lyksdlewlr gigwsplgsl eaeknkrase iisekkyrqp pdrnkftsip 2461 damdivlakt naknrsdrly reawdkdktq ihimpdtpdi vlakanlint sdklyrmgye 2521 elkrkgydlp vdaipikaak asreiaseyk ykegfrkqlg hhigarnied dpkmmwsmhv 2581 akiqsdreyk kdfekwktkf sspvdmlgvv lakkcqtlvs dvdyknylhq wtclpdqsdv 2641 iharqaydlq sdnlyksdlq wlkgigwmts gsledeknkr atqilsdhvy rqhpdqfkfs 2701 slmdsipmvl aknnaitmnh rlyteawdkd kttvhimpdt pevllakqnk vnyseklykl 2761 gleeakrkgy dmrvdaipik aakasrdias efkykegyrk qlghhigara irddpkmmws 2821 mhvakiqsdr eykkdfekwk tkfsspvdml gvvlakkcqt lvsdvdykny lhqwtclpdq 2881 sdviharqay dlqsdnmyks dlqwmrgigw vsigsldvek ckrateilsd kiyrqppdrf 2941 kftsvtdsle qvlaknnait mnkrlyteaw dkdktqihim pdtpeimlar mnkinysesl 3001 yklaneeakk kgydlrsdai pivaakasrd iisdykykdg yckqlghhig arnieddpkm 3061 mwsmhvakiq sdreykkdfe kwktkfsspv dmlgvvlakk cqtlvsdvdy knylhewtcl 3121 pdqsdvihar qaydlqsdni yksdlqwlrg igwvpigsmd vvkckratei lsdniyrqpp 3181 dklkftsvtd sleqvlaknn alnmnkrlyt eawdkdktqi himpdtpeim larqnkinys 3241 etlyklanee akkkgydlrs daipivaaka srdvisdyky kdgyrkqlgh higarniedd 3301 pkmmwsmhva kiqsdreykk dfekwktkfs spvdmlgvvl akkcqtlvsd vdyknylhew 3361 tclpdqndvi harqaydlqs dniyksdlqw lrgigwvpig smdvvkckra aeilsdniyr 3421 qppdklkfts vtdsleqvla knnalnmnkr lyteawdkdk tqvhimpdtp eimlarqnki 3481 nyseslyrqa meeakkegyd lrsdaipiva akasrdiasd ykykeayrkq lghhigarav 3541 hddpkimwsl hiakvqsdre ykkdfekykt rysspvdmlg ivlakkcqtl vsdvdykhpl 3601 hewiclpdqn diiharkayd lqsdnlyksd lewmkgigwv pidslevvra kragellsdt 3661 iyrqrpetlk ftsitdtpeq vlaknnalnm nkrlyteawd ndkktihvmp dtpeimlakl 3721 nrinysdkly klaleeskke gydlrldaip iqaakasrdi asdykykegy rkqlghhiga 3781 rnikddpkmm wsihvakiqs dreykkefek wktkfsspvd mlgvvlakkc qilvsdidyk 3841 hplhewtclp dqndviqark aydlqsdaiy ksdlewlrgi gwvpigsvev ekvkrageil 3901 sdrkyrqpad qlkftcitdt peivlaknna ltmskhlyte awdadktsih vmpdtpdill 3961 aksnsanisq klytkgwdes kmkdydlrad aisiksakas rdiasdykyk eayekqkghh 4021 igaqsieddp kimcaihagk iqsereykke fqkwktkfss pvdmlsilla kkcqtlvtdi 4081 dyrnylhewt cmpdqndiiq akkaydlqsd svykadlewl rgigwmpegs vemnrvkvaq 4141 dlvnerlyrt rpealsftsi vdtpevvlak anslqisekl yqeawnkdks nitipsdtpe 4201 mlqahinalq isnklyqkdw ndakqkgydi radaieikha kasreiasey kykegyrkql 4261 ghhmgfrtlq ddpksvwaih aakiqsdrey kkayekskgi hntpldmmsi vqakkcqvlv 4321 sdidyrnylh qwtclpdqnd viqakkaydl qsdnlyksdl ewlkgigwlp egsvevmrvk 4381 naqnllnerl yrikpealkf tsivdtpevi qakinavqis eplyrdawek ekanvnvpad 4441 tplmlqskin alqisnkryq qawedvkmtg ydlradaigi qhakasrdia sdylyktaye 4501 kqkghyigcr sakedpklvw aanvlkmqnd rlykkayndh kakisipvdm vsisaakegq 4561 alasdvdyrh ylhhwscfpd qndviqarka ydlqsdsvyk adlewlrgig wmpegsvemn 4621 rvkvaqdlvn erlyrtrpea lsftsivdtp evvlakansl qiseklyqea wnkdksniti 4681 psdtpemlqa hinalqisnk lyqkdwndtk qkgydirada ieikhakasr eiaseykyke 4741 gyrkqlghhm gfrtlqddpk svwaihaaki qsdreykkay ekskgihntp ldmmsivqak 4801 kcqvlvsdid yrnylhqwtc lpdqndviqa kkaydlqsdn lyksdlewlk gigwlpegsv 4861 evmrvknaqn llnerlyrik pealkftsiv dtpeviqaki navqiseply rnawekekan 4921 vnvpadtplm lqskinalqi snkryqqawe dvkmtgydlr adaigiqhak asrdiasdyl 4981 yktayekqkg hyigcrsake dpklvwaanv lkmqndrlyk kayndhkaki sipvdmvsis 5041 aakegqalas dvdyrhylhh wscfpdqndv iqarkaydlq sdsvykadle wlrgigwmpe 5101 gsvemnrvkv aqdlvnerly rtrpealsft sivdtpevvl akanslqise klyqeawnkd 5161 ksnitipsdt pemlqahina lqisnklyqk dwndtkqkgy diradaieik hakasreias 5221 eykykegyrk qlghhmgfrt lqddpksvwa ihaakiqsdr eykkayeksk gihntpldmm 5281 sivqakkcqv lvsdidyrny lhqwtclpdq ndviqakkay dlqsdnlyks dlewlkgigw 5341 lpegsvevmr vknaqnllne rlyrikpeal kftsivdtpe viqakinavq iseplyrdaw 5401 ekekanvnvp adtplmlqsk inalqisnkr yqqawedvkm tgydlradai giqhakasrd 5461 iasdylykta yekqkghyig crsakedpkl vwaanvlkmq ndrlykkayn dhkakisipv 5521 dmvsisaake gqalasdvdy rhylhrwscf pdqndviqar kaydlqsdal ykadlewlrg 5581 igwmpqgspe vlrvknaqni fcdsvyrtpv vnlkytsivd tpevvlaksn aenisipkyr 5641 evwdkdktsi himpdtpein laranalnvs nklyregwde mkagcdvrld aipiqaakas 5701 reiasdykyk ldhekqkghy vgtltarddn kirwaliadk lqnereyrld wakwkakiqs 5761 pvdmlsilhs knsqalvsdm dyrnylhqwt cmpdqndviq akkayelqsd nvykadlewl 5821 rgigwmpnds vsvnhakhaa difsekkyrt kietlnftpv ddrvdyvtak qsgeilddik 5881 yrkdwnatks kytltetpll htaqeaaril dqylykegwe rqkatgyilp pdavpfvhah 5941 hcndvqselk ykaehvkqkg hyvgvptmrd dpklvwfeha gqiqnerlyk edyhktkaki 6001 nipadmvsvl aakqgqtlvs didyrnylhq wmchpdqndv iqarkaydlq sdnvyradle 6061 wlrgigwipl dsvdhvrvtk nqemmsqiky kknalenypn frsvvdppei vlakinsvnq 6121 sdvkyketfn kakgkytfsp dtphishskd mgklystily kgawegtkay gytlderyip 6181 ivgakhadlv nselkykety ekqkghylag kvigefpgvv hcldfqkmrs alnyrkhyed 6241 tkanvhipnd mmnhvlakrc qyilsdleyr hyfhqwtsll eepnvirvrn aqeilsdnvy 6301 kddlnwlkgi gcyvwdtpqi lhakksydlq sqlqytaagk enlqnynlvt dtplyvtavq 6361 sginasevky kenyhqikdk yttvletvdy drtrnlknly ssnlykeawd rvkatsyilp 6421 sstlslthak nqkhlashik yreeyekfka lytlprsvdd dpntarclrv gklnidrlyr 6481 svyeknkmki hivpdmvemv takdsqkkvs eidyrlrlhe wichpdlqvn dhvrkvtdqi 6541 sdivykddln wlkgigcyvw dtpeilhakh aydlrddiky kahmlktrnd yklvtdtpvy 6601 vqavksgkql sdavyhydyv hsvrgkvapt tktvdldral hayklqssnl yktslrtlpt 6661 gyrlpgdtph fkhikdtrym ssyfkykeay ehtkaygytl gpkdvpfvhv rrvnnvtser 6721 lyrelyhklk dkihttpdtp eirqvkktqe avseliyksd ffkmqghmis lpytpqvihc 6781 ryvgditsdi kykedlqvlk gfgcflydtp dmvrsrhlrk lwsnylytdk arkmrdkykv 6841 vldtpeyrkv qelkthlsel vyraagkkqk siftsvpdtp dllrakrgqk lqsqylyvel 6901 atkerphhha gnqttalkha kdvkdmvsek kykiqyekmk dkytpvpdtp ilirakrayw 6961 nasdlryket fqktkgkyht vkdaldivyh rkvtddiski kykenymsql giwrsipdrp 7021 ehfhhravtd tvsdvkyked ltwlkgigcy aydtpdftla eknktlysky kykevfertk 7081 sdfkyvadsp inrhfkyatq lmnekkyrad yeqrkdkyhl vvdeprhlla ktagdqisqi 7141 kyrknyeksk dkftsivdtp ehlrttkvnk qisdilykle ynkakprgyt tihdtpmllh 7201 vrkvkdevsd lkykevyqrn ksnctiepda vhikaakday kvntnldykk qyeankahwk 7261 wtpdrpdflq aaksslqqsd feykldrefl kgcklsvtdd kntvlalrnt liesdlkyke 7321 khvkergtch avpdtpqill aktvsnlvse nkykdhvkkh laqgsyttlp etrdtvhvke 7381 vtkhvsdtny kkkfvkekgk snysimlepp evkhamevak kqsdvayrkd akenlhyttv 7441 adrpdikkat qaakqaseve yrakhrkegs hglsmlgrpd iemakkaakl ssqvkyrenf 7501 dkekgktpky npkdsqlykv mkdannlase vkykadlkkl hkpvtdmkes limnhvlnts 7561 qlassyqykk kyekskghyh tipdnleqlh lkeatelqsi vkykekyeke rgkpmldfet 7621 ptyitakesq qmqsgkeyrk dyeesikgrn ltglevtpal lhvkyatkia sekeyrkdle 7681 esirgkglte medtpdmlra knatqilnek eykrdlelev kgrglnaman etpdfmrarn 7741 atdiasqiky kqsaemekan ftsvvdtpei ihaqqvknls sqkkykedae ksmsyyetvl 7801 dtpeiqrvre nqknfsllqy qcdlknskgk itvvqdtpei lrvkenqknf ssvlykedvs 7861 pgtaigktpe mmrvkqtqdh issvkykeai gqgtpipdlp evkrvketqk hissvmyken 7921 lgtgipttvt peiervkrnq enfssvlyke nlgkgiptpi tpemervkrn qenfssilyk 7981 enlskgtplp vtpemervkl nqenfssvly kenvgkgipi pitpemervk hnqenfssvl 8041 ykenlgtgip ipitpemqrv khnqenlssv lykenmgkgt plpvtpemer vkhnqeniss 8101 vlykenmgkg tplpvtpeme rvkhnqenis svlykenmgk gtplavtpem ervkhnqeni 8161 ssvlykenvg katatpvtpe mqrvkrnqen issvlykenl gkatptpftp emervkrnqe 8221 nfssvlyken mrkatptpvt pemerakrnq enissvlysd sfrkqiqgka ayvldtpemr 8281 rvretqrhis tvkyhedfek hkgcftpvvt dpitervkkn mqdfsdinyr giqrkvveme 8341 qkrndqdqet itglrvwrtn pgsvfdydpa edniqsrslh minvqaqrrs reqsrsasal 8401 sisggeekse hseapdhhls tysdggvfav staykhaktt elpqqrsssv atqqttvssi 8461 pshpstagki framydymaa dadevsfkdg daiinvqaid egwmygtvqr tgrtgmlpan 8521 yveai';

}

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (1.856378206e+29 + 0);

cursor2x = spawnerx - (1.23758547e+29 - 9.28189103e+27);

repeat\_endcell1 = Math.round(15 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(11 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 9.28189103e+27) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

} else if (randomizer <= 50) {

peptidesequence = '1 mviqkekksc gqvveewkef vwnprthqfm grtgtswafi llfylvfygf ltamftltmw 61 vmlqtvsdht pkyqdrlatp glmirpkten ldvivnvsdt eswdqhvqkl nkflepynds 121 iqaqkndvcr pgryyeqpdn gvlnypkrac qfnrtqlgnc sgigdsthyg ystgqpcvfi 181 kmnrvinfya ganqsmnvtc agkrdedaen lgnfvmfpan gnidlmyfpy ygkkfhvnyt 241 qplvavkfln vtpnvevnve crinaaniat dderdkfagr vafklrinkt';

} else if (randomizer <= 75) {

peptidesequence = '1 mtknekksln qslaewklfi ynpttgeflg rtakswglil lfylvfygfl aalfsftmwv 61 mlqtlndevp kyrdqipspg lmvfpkpvta leytfsrsdp tsyagyiedl kkflkpytle 121 eqknltvcpd galfeqkgpv yvacqfpisl lqacsgmndp dfgysqgnpc ilvkmnriig 181 lkpegvprid cvsknedipn vavyphngmi dlkyfpyygk klhvgylqpl vavqvsfapn 241 ntgkevtvec kidgsanlks qddrdkflgr vmfkitara';

} else {

peptidesequence = ' 1 mrrqlrsrra psfpysyryr lddpdeanqn yladeeeeae eearvtvvpk seeeeeeeek 61 eeeeeeekee eegqgqptgn awwqklqims eylwdperrm flartgqsws lilliyfffy 121 aslaavitlc mytlfltisp yiptftervk ppgvmirpfa hslnfnfnvs epdtwqhyvi 181 slngflqgyn dslqeemnvd cppgqyfiqd gnededkkac qfkrsflknc sgledptfgy 241 stgqpcillk mnrivgfrpe lgdpvkvsck vqrgdendir sisyypesas fdlryypyyg 301 klthvnytsp lvamhftdvv knqavpvqcq lkgkgvindv indrfvgrvi ftlniet';

}

protein();

} else if (randomizer <= 75) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 medghsktve qslnffgtdp ergltldqik anqkkygpne lpteegksiw qlvleqfddl 61 lvkilllaai isfvlalfee heetftafve plvillilia navvgvwqer naesaiealk 121 eyepemgkvv rqdksgiqkv rakeivpgdl vevsvgdkip adirithiys ttlridqsil 181 tgesvsvikh tdaipdprav nqdkknilfs gtnvaagkar gvvigtglst aigkirtems 241 eteeiktplq qkldefgeql skvisvicva vwainighfn dpahggswik gaiyyfkiav 301 alavaaipeg lpavittcla lgtrrmakkn aivrslpsve tlgctsvics dktgtlttnq 361 msvsrmfifd kvegndssfl efemtgstye pigevflngq rikaadydtl qelsticimc 421 ndsaidynef kqafekvgea tetalivlae klnsfsvnks gldrrsaaia crgeietkwk 481 keftlefsrd rksmssyctp lkasrlgtgp klfvkgapeg vlerctharv gttkvpltsa 541 lkakilaltg qygtgrdtlr clalavadsp mkpdemdlgd stkfyqyevn ltfvgvvgml 601 dpprkevfds ivrcraagir vivitgdnka taeaicrrig vfaededttg ksysgrefdd 661 lspteqkaav arsrlfsrve pqhkskivef lqsmneisam tgdgvndapa lkkaeigiam 721 gsgtavaksa aemvladdnf ssivsaveeg raiynnmkqf irylissnig evvsifltaa 781 lglpealipv qllwvnlvtd glpatalgfn ppdldimekp prkadeglis gwlffrymai 841 gfyvgaatvg aaawwfvfsd egpklsywql thhlsclggg defkgvdcki fsdphamtma 901 lsvlvtieml namnslsenq slitmppwcn lwligsmals ftlhfvilyv dvlstvfqvt 961 plsaeewitv mkfsipvvll detlkfvark iadvpdvvvd rm';

protein();

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = '1 mgevtaeeve kfldsnigfa kqyynlhyra klisdllgak eaavdfsnyh spssmeesei 61 ifdllrdfqe nlqtekcifn vmkklcfllq adrmslfmyr trngiaelat rlfnvhkdav 121 ledclvmpdq eivfpldmgi vghvahskki anvpnteede hfcdfvdilt eyktknilas 181 pimngkdvva iimavnkvdg shftkrdeei llkylnfanl imkvyhlsyl hncetrrgqi 241 llwsgskvfe eltdierqfh kalytvrafl ncdrysvgll dmtkqkeffd vwpvlmgevp 301 pysgprtpdg reinfykvid yilhgkedik vipnpppdhw alvsglpayv aqnglicnim 361 napaedffaf qkepldesgw miknvlsmpi vnkkeeivgv atfynrkdgk pfdemdetlm 421 esltqflgws vlnpdtyesm nklenrkdif qdivkyhvkc dneeiqkilk trevygkepw 481 eceeeelaei lqaelpdadk yeinkfhfsd lpltelelvk cgiqmyyelk vvdkfhipqe 541 alvrfmysls kgyrkityhn wrhgfnvgqt mfsllvtgkl kryftdleal amvtaafchd 601 idhrgtnnly qmksqnplak lhgssilerh hlefgktllr deslnifqnl nrrqhehaih 661 mmdiaiiatd lalyfkkrtm fqkivdqskt yeseqewtqy mmleqtrkei vmammmtacd 721 lsaitkpwev qsqvallvaa efweqgdler tvlqqnpipm mdrnkadelp klqvgfidfv 781 ctfvykefsr fheeitpmld gitnnrkewk aladeydakm kvqeekkqkq qsaksaaagn 841 qpggnpspgg attskscciq';

} else if (randomizer <= 40) {

peptidesequence = '1 mslseeqars fldqnpdfar qyfgkklspe nvaaacedgc ppdcdslrdl cqveestall 61 elvqdmqesi nmervvfkvl rrlctllqad rcslfmyrqr ngvaelatrl fsvqpdsvle 121 dclvppdsei vfpldigvvg hvaqtkkmvn vedvaecphf ssfadeltdy ktknmlatpi 181 mngkdvvavi mavnklngpf ftsededvfl kylnfatlyl kiyhlsylhn cetrrgqvll 241 wsankvfeel tdierqfhka fytvraylnc erysvglldm tkekeffdvw svlmgesqpy 301 sgprtpdgre ivfykvidyv lhgkeeikvi ptpsadhwal asglpsyvae sgficnimna 361 sademfkfqe galddsgwli knvlsmpivn kkeeivgvat fynrkdgkpf deqdevlmes 421 ltqflgwsvm ntdtydkmnk lenrkdiaqd mvlyhvkcdr deiqlilptr arlgkepadc 481 dedelgeilk eelpgpttfd iyefhfsdle cteldlvkcg iqmyyelgvv rkfqipqevl 541 vrflfsiskg yrrityhnwr hgfnvaqtmf tllmtgklks yytdleafam vtaglchdid 601 hrgtnnlyqm ksqnplaklh gssilerhhl efgkfllsee tlniyqnlnr rqhehvihlm 661 diaiiatdla lyfkkramfq kivdesknyq dkkswveyls lettrkeivm ammmtacdls 721 aitkpwevqs kvallvaaef weqgdlertv ldqqpipmmd rnkaaelpkl qvgfidfvct 781 fvykefsrfh eeilpmfdrl qnnrkewkal adeyeakvka leekeeeerv aakkvgteic 841 nggpapksst ccil';

} else if (randomizer <= 60) {

peptidesequence = '1 mgeinqvave kyleenpqfa keyfdrklrv evlgeifkns qvpvqssmsf seltqveesa 61 lclellwtvq eeggtpeqgv hralqrlahl lqadrcsmfl crsrngipev asrlldvtpt 121 skfednlvgp dkevvfpldi givgwaahtk kthnvpdvkk nshfsdfmdk qtgyvtknll 181 atpivvgkev lavimavnkv nasefskqde evfskylnfv siilrlhhts ymyniesrrs 241 qilmwsankv feeltdverq fhkalytvrs ylncerysig lldmtkekef ydewpiklge 301 vepykgpktp dgrevnfyki idyilhgkee ikviptppad hwtlisglpt yvaengficn 361 mmnapadeyf tfqkgpvdet gwviknvlsl pivnkkediv gvatfynrkd gkpfdehdey 421 itetltqflg wsllntdtyd kmnklenrkd iaqemlmnqt katpeeiksi lkfqeklnvd 481 viddceekql vailkedlpd prsaelyefr fsdfpltehg likcgirlff einvvekfkv 541 pvevltrwmy tvrkgyravt yhnwrhgfnv gqtmftllmt grlkkyytdl eafamlaaaf 601 chdidhrgtn nlyqmkstsp larlhgssil erhhleyskt llqdeslnif qnlnkrqfet 661 vihlfevaii atdlalyfkk rtmfqkivda ceqmqteeea ikyvtvdptk keiimammmt 721 acdlsaitkp wevqsqvalm vanefweqgd lertvlqqqp ipmmdrnkrd elpklqvgfi 781 dfvctfvyke fsrfhkeitp mlsglqnnrv ewksladeyd akmkvieeea kkqeggaeka 841 aedsgggddk ksktclml';

} else if (randomizer <= 80) {

peptidesequence = '1 msdnttlpap asnqgpttpr kgppkfkqrq trqfkskppk kgvkgfgddi pgmeglgtdi 61 tvicpweafs hlelhelaqf gii';

} else if (randomizer <= 100) {

peptidesequence = '1 mnleppkaef rsatrvaggp vtprkgppkf kqrqtrqfks kppkkgvqgf gddipgmegl 61 gtditvicpw eafnhlelhe laqygii';

}

protein();

}

} else {

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1.856378206e+29;

cursor2y = spawnery - (1.856378206e+29 + 0);

cursor2x = spawnerx - (1.23758547e+29 + 1.23758547e+28);

repeat\_endcell1 = Math.round(15 - 0);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(11 - 0);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 4.950341882e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 4.950341882e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_sarcomere\_protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor2y = cursor2y - 6.187927353e+28;

}

cursor2y = cursor2y + 6.187927353e+28 \* 16;

cursor2x = cursor2x - 6.187927353e+28 \* 1;

}

cursor2z = spawnerz + 1;

cursor2y = spawnery - (2.351412394e+29 + (6.187927353e+28 \* 12 + 1.23758547e+28));

cursor2x = spawnerx - 6.806720089e+28;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = cursor2x - 1;

cursor1y = cursor2y - 1;

cursor1z = cursor2z + 1.856378206e+28;

}

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((1 \* 1.113826923e+30) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 150);

if (randomizer <= 50) {

peptidesequence = ' 10 20 30 40 50 MGLSDGEWQL VLNVWGKVEA DIPGHGQEVL IRLFKGHPET LEKFDKFKHL 60 70 80 90 100 KSEDEMKASE DLKKHGATVL TALGGILKKK GHHEAEIKPL AQSHATKHKI 110 120 130 140 150 PVKYLEFISE CIIQVLQSKH PGDFGADAQG AMNKALELFR KDMASNYKEL GFQG ';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mrmsvglsll lplsgrtfll llsvvmaqsh wpsepseavr dwenqleasm hsvlsdlhea 61 vptvvgipdg tavvgrsfrv tiptdliass gdiikvsaag kealpswlhw dsqshtlegl 121 pldtdkgvhy isvsatrlga ngshipqtss vfsievyped hselqsvrta spdpgevvss 181 acaadepvtv ltvildadlt kmtpkqridl lhrmrsfsev elhnmklvpv vnnrlfdmsa 241 fmagpgnakk vvengallsw klgcslnqns vpdihgveap aregamsaql gypvvgwhia 301 nkkpplpkrv rrqihatptp vtaigpptta iqeppsrivp tptspaiapp tetmappvrd 361 pvpgkptvti rtrgaiiqtp tlgpiqptrv seagttvpgq irptmtipgy veptavatpp 421 ttttkkprvs tpkpatpstd stttttrrpt kkprtprpvp rvttkvsitr letaspptri 481 rtttsgvprg gepnqrpelk nhidrvdawv gtyfevkips dtfydhedtt tdklkltlkl 541 reqqlvgeks wvqfnsnsql myglpdsshv gkheyfmhat dkgglsavda feihvhrrpq 601 gdraparfka kfvgdpalvl ndihkkialv kklafafgdr ncstitlqni trgsivvewt 661 nntlplepcp keqiaglsrr iaeddgkprp afsnalepdf katsitvtgs gscrhlqfip 721 vvpprrvpse apptevpdrd peksseddvy lhtvipavvv aailliagii amicyrkkrk 781 gkltledqat fikkgvpiif adelddskpp psssmplilq eekaplpppe ypnqsvpett 841 plnqdtmgey tplrdedpna ppyqppppft apmegkgsrp knmtpyrspp pyvpp';

protein();

} else {

heme();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

intercalated\_disks();

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 1.23758547e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.249806627e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 1.113826923e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 2.475170941e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.373565174e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 3.712756412e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - 6.497323721e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((36 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = ' 1 mrrqlrsrra psfpysyryr lddpdeanqn yladeeeeae eearvtvvpk seeeeeeeek 61 eeeeeeekee eegqgqptgn awwqklqims eylwdperrm flartgqsws lilliyfffy 121 aslaavitlc mytlfltisp yiptftervk ppgvmirpfa hslnfnfnvs epdtwqhyvi 181 slngflqgyn dslqeemnvd cppgqyfiqd gnededkkac qfkrsflknc sgledptfgy 241 stgqpcillk mnrivgfrpe lgdpvkvsck vqrgdendir sisyypesas fdlryypyyg 301 klthvnytsp lvamhftdvv knqavpvqcq lkgkgvindv indrfvgrvi ftlniet';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 6.187927353e+26);

cursor1y = spawnery - 1 \* 6.187927353e+28;

cursor1x = spawnerx - (1.175706197e+30 - 4.950341882e+27);

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((1 \* 1.113826923e+30) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 5.754772438e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function default\_cardiomyocyte\_2() {

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.237523591e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_3 = Math.round(1.23758547e+30 / density\_parameter\_x - 2);

generation\_parameter\_z = Math.round(6.187927353e+30 / density\_parameter\_x - 2);

}

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

set\_earth\_rotation();

if ((cursor1x > spawnerx + 1.175706197e+30 && cursor1x < spawnerx + 6.187927353e+28) && (cursor1y > spawnery + 1.175706197e+30 && cursor1y < spawnery + 6.187927353e+28)) {

randomize\_cytosol();

} else {

randomize\_ECF();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function audiocyte() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + (2.784567309e+30 + eara), spawnery - 6.187927352e+29, spawnerx - 6.187927352e+29];

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

if (make == true) {

lipid2 = 1.856378206e+30;

lipid1 = 5.878530986e+29;

cursor2z = spawnerz + 10 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

if (unused\_variable == 5) {

cursor2z = spawnerz + 1 \* 2.165774573e+30;

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 9.28189103e+28;

lipid2 = 5.569134618e+29;

lipid1 = 6.187927353e+28;

for (countcell1 = 0; countcell1 < 4; countcell1++) {

lipid\_wall\_z();

cursor2x = cursor2x - 1 \* 1.23758547e+29;

}

cursor2y = spawnery - 1 \* 9.28189103e+28;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid2 = 6.187927353e+28;

lipid1 = 5.569134618e+29;

for (countcell1 = 0; countcell1 < 4; countcell1++) {

lipid\_wall\_z();

cursor2y = cursor2y - 1 \* 1.23758547e+29;

}

cursor2z = spawnerz + 1 \* 2.165774573e+30;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 3.093963676e+28;

earb = 1;

for (countcell1 = 0; countcell1 < 5; countcell1++) {

if (unused\_variable == 5) {

lipid2 = 6.187927353e+28 \* earb + eara;

lipid1 = 6.187927353e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2x = cursor2x - 0.5 \* 1.23758547e+29;

lipid\_wall\_x();

cursor2x = cursor2x + 0.5 \* 1.23758547e+29;

cursor2y = cursor2y - 0.5 \* 1.23758547e+29;

lipid\_wall\_y();

cursor2y = cursor2y + 0.5 \* 1.23758547e+29;

cursor2z = cursor2z + (6.187927353e+28 \* earb + eara);

lipid2 = lipid1;

lipid\_wall\_z();

cursor2z = cursor2z - (6.187927353e+28 \* earb + eara);

}

cursor2x = cursor2x - 1 \* 1.23758547e+29;

for (countcell2 = 0; countcell2 < 4; countcell2++) {

earb = earb + 1;

if (unused\_variable == 5) {

lipid2 = 6.187927353e+28 \* earb + eara;

lipid1 = 6.187927353e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2x = cursor2x - 0.5 \* 1.23758547e+29;

lipid\_wall\_x();

cursor2x = cursor2x + 0.5 \* 1.23758547e+29;

cursor2y = cursor2y - 0.5 \* 1.23758547e+29;

lipid\_wall\_y();

cursor2y = cursor2y + 0.5 \* 1.23758547e+29;

cursor2z = cursor2z + (6.187927353e+28 \* earb + eara);

lipid2 = lipid1;

lipid\_wall\_z();

cursor2z = cursor2z - (6.187927353e+28 \* earb + eara);

if (unused\_variable == 5) {

cursor1z = cursor2z;

cursor1y = cursor2y;

cursor1x = cursor2x;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((2 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((2 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((6.187927353e+28 \* earb + eara) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 60);

if (earb == 9) {

if (randomizer <= 15) {

peptidesequence = '1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

} else if (randomizer <= 30) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 30) {

peptidesequence = '1 mfrqfylwtc lasgiilgsl feiclgqydd dwqyedckla rggppativa ideesrngti 61 lvdnmlikgt aggpdptiel slkdnvdywv lmdpvkqmlf lnstgrvldr dppmnihsiv 121 vqvqcinkkv gtiiyhevri vvrdrndnsp tfkhesyyat vneltpvgtt iftgfsgdng 181 atdiddgpng qieyviqynp ddptsndtfe iplmltgniv lrkrlnyedk tryfviiqan 241 draqnlnerr tttttltvdv ldgddlgpmf lpcvlvpntr dcrpltyqaa ipelrtpeel 301 npiivtppiq aidqdrniqp psdrpgilys ilvgtpedyp rffhmhprta elsllepvnr 361 dfhqkfdlvi kaeqdnghpl pafaglhiei ldennqspyf tmpsyqgyil esapvgatis 421 dslnltsplr ivaldkdied tkdpelhlfl ndytsvftvt qtgitryltl lqpvdreeqq 481 tytfsitafd gvqesepviv niqvmdandn tptfpeisyd vyvytdmrpg dsviqltavd 541 adegsngeit yeilvgaqgd fiinkttgli tiapgvemiv grtyaltvqa adnappaerr 601 nsictvyiev lppnnqsppr fpqlmyslei seamrvgavl lnlqatdreg dsityaieng 661 dpqrvfnlse ttgiltlgka ldrestdryi liitasdgrp dgtstatvni vvtdvndnap 721 vfdpylprnl svveeeanaf vgqvkatdpd agingqvhys lgnfnnlfri tsngsiytav 781 klnrevrdyy elvvvatdga vhprhstltl aikvldiddn spvftnstyt vlveenlpag 841 ttilqieakd vdlganvsyr irspevkhff alhpftgels llrsldyeaf pdqeasitfl 901 veafdiygtm ppgiatvtvi vkdmndyppv fskriykgmv apdavkgtpi ttvyaedadp 961 pglpasrvry rvddvqfpyp asifeveeds grvitrvnln eepttifklv vvafddgepv 1021 msssatvkil vlhpgeiprf tqeeyrpppv selatkgtmv gvisaaainq sivysivsgn 1081 eedtfginni tgviyvngpl dyetrtsyvl rvqadslevv lanlrvpsks ntakvyieiq 1141 dennhppvfq kkfyiggvse darmftsvlr vkatdkdtgn ysvmayrlii ppikegkegf 1201 vvetytglik tamlfhnmrr syfkfqviat ddygkglsgk advlvsvvnq ldmqvivsnv 1261 pptlvekkie dlteildryv qeqipgakvv vesigarrhg dafsledytk cdltvyaidp 1321 qtnraidrne lfkfldgkll dinkdfqpyy geggrileir tpeavtsikk rgeslgyteg 1381 allalafiii lccipailvv lvsyrqfkvr qaectktari qaalpaakpa vpapapvaap 1441 ppppppppga hlyeelgdss mhnlfllyhf qqsrgnnsvs edrkhqqvvm pfssntieah 1501 ksahvdgslk snklksarkf tflsdeddls ahnplykeni sqvstnsdis qrtdfvdpfs 1561 pkiqaksksl rgprekiqrl wsqsvslprr lmrkvpnrpe iidlqqwqgt rqkaenentg 1621 ictnkrgssn plltteeanl tekeeirqge tlmiegteql kslssdssfc fprphfsfst 1681 lptvsrtvel ksepnvissp aecslelsps rpcvlhssls rretpicmlp ieternifen 1741 fahppnisps acplpppppi sppspppapa plapppdisp fslfcpppsp psiplplppp 1801 tffplsvsts gpptppllpp fptplppppp sipcppppsa sflstecvci tgvkcttnlm 1861 paekikssmt qlstttvckt dpqrepkgil rhvknlaele ksvanmysqi eknylrtnvs 1921 elqtmcpsev tnmeitseqn kgslnniveg tekqshsqst sl';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgrhvatsch vawllvlisg cwgqvnrlpf ftnhffdtyl lisedtpvgs svtqllaqdm 61 dndplvfgvs geeasrffav epdtgvvwlr qpldretkse ftvefsvsdh qgvitrkvni 121 qvgdvndnap tfhnqpysvr ipentpvgtp ifivnatdpd lgaggsvlys fqppsqffai 181 dsargivtvi reldyettqa yqltvnatdq dktrplstla nlaiiitdvq dmdpifinlp 241 ystniyehsp pgttvriita idqdkgrprg igytivsgnt nsifaldyis gvltlnglld 301 renplyshgf iltvkgteln ddrtpsdatv tttfnilvid indnapefns seysvaitel 361 aqvgfalplf iqvvdkdenl glnsmfevyl vgnnshhfii sptsvqgkad irirvaipld 421 yetvdrydfd lfanesvpdh vgyakvkitl inendnrpif sqplynisly envtvgtsvl 481 tvlatdndag tfgevsyffs ddpdrfsldk dtglimliar ldyeliqrft ltiiardggg 541 eettgrvrin vldvndnvpt fqkdayvgal renepsvtql vrlratdeds ppnnqitysi 601 vsasafgsyf dislyegygv isvsrpldye qisngliylt vmamdagnpp lnstvpvtie 661 vfdendnppt fskpayfvsv venimagatv lflnatdldr sreygqesii yslegstqfr 721 inarsgeitt tslldretks eyilivravd ggvghnqktg iatvnitlld indnhptwkd 781 apyyinlvem tppdsdvttv vavdpdlgen gtlvysiqpp nkfyslnstt gkirtthaml 841 drenpdphea elmrkivvsv tdcgrpplka tssatvfvnl ldlndndptf qnlpfvaevl 901 egipagvsiy qvvaidldeg lnglvsyrmp vgmprmdfli nsssgvvvtt teldreriae 961 yqlrvvasda gtptksstst ltihvldvnd etptffpavy nvsvsedvpr efrvvwlnct 1021 dndvglnael syfitggnvd gkfsvgyrda vvrtvvgldr ettaaymlil eaidngpvgk 1081 rhtgtatvfv tvldvndnrp iflqssyeas vpedipeghs ilqlkatdad egefgrvwyr 1141 ilhgnhgnnf rihvsngllm rgprpldrer nsshvlivea ynhdlgpmrs svrvivyved 1201 indeapvftq qqysrlglre tagigtsviv vqatdrdsgd gglvnyrils gaegkfeide 1261 stgliitvny ldyetktsym mnvsatdqap pfnqgfcsvy itllneldea vqfsnasyea 1321 ailenlalgt eivrvqaysi dnlnqityrf naytstqaka lfkidaitgv itvqglvdre 1381 kgdfytltvv addggpkvds tvkvyitvld endnsprfdf tsdsavsipe dcpvgqrvat 1441 vkawdpdags ngqvvfslas gniagafeiv ttndsigevf varpldreel dhyilqvvas 1501 drgtpprkkd hilqvtildi ndnppviesp fgynvsvnen vgggtavvqv ratdrdigin 1561 svlsyyiteg nkdmafrmdr isgeiatrpa ppdrerqsfy hlvatvedeg tptlsatthv 1621 yvtivdendn apmfqqphye vlldegpdtl ntslitiqal dldegpngtv tyaivagniv 1681 ntfridrhmg vitaakeldy eishgrytli vtatdqcpil shrltstttv lvnvndindn 1741 vptfprdyeg pfevtegqpg prvwtflahd rdsgpngqve ysimdgdplg efvispvegv 1801 lrvrkdveld retiafynlt icardrgmpp lsstmlvgir vldindndpv llnlpmniti 1861 senspvssfv ahvlasdads gcnarltfni tagnreraff inattgivtv nrpldrerip 1921 eykltisvkd npenpriarr dydlllifls dendnhplft kstyqaevme nspagtpltv 1981 lngpilalda dqdiyavvty qllgaqsglf dinsstgvvt vrsgviidre afsppilell 2041 llaediglln stahllitil ddndnrptfs patltvhlle ncppgfsvlq vtatdedsgl 2101 ngelvyriea gaqdrflihl vtgvirvgna tidreeqesy rltvvatdrg tvplsgtaiv 2161 tiliddinds rpeflnpiqt vsvlesaepg tvianitaid hdlnpkleyh ivgivakddt 2221 drlvpnqeda favnintgsv mvkspmnrel vatyevtlsv idnasdlper svsvpnaklt 2281 vnvldvndnt pqfkpfgity ymerilegat pgttliavaa vdpdkglngl vtytlldlvp 2341 pgyvqledss agkvianrtv dyeevhwlnf tvrasdngsp praaeipvyl eivdindnnp 2401 ifdqpsyqea vfedvpvgti iltvtatdad sgnfalieys lgdgeskfai npttgdiyvl 2461 ssldrekkdh yiltalakdn pgdvasnrre nsvqvviqvl dvndcrpqfs kpqfstsvye 2521 nepagtsvit mmatdqdegp ngeltysleg pgveafhvdm dsglvttqrp lqsyekfslt 2581 vvatdggepp lwgttmllve vidvndnrpv fvrppngtil hireeiplrs nvyevyatdk 2641 deglngavry sflktagnrd weffiidpis gliqtaqrld resqavysli lvasdlgqpv 2701 pyetmqplqv alediddnep lfvrppkgsp qyqlltvpeh sprgtlvgnv tgavdadegp 2761 naivyyfiaa gneeknfhlq pdgcllvlrd ldrereaifs fivkassnrs wtpprgpspt 2821 ldlvadltlq evrvvledin dqpprftkae ytagvatdak vgseliqvla ldadignnsl 2881 vfysilaihy fralandsed vgqvftmgsm dgilrtfdlf mayspgyfvv divardlagh 2941 ndtaiigiyi lrddqrvkiv ineipdrvrg feeefihlls nitgaivntd nvqfhvdkkg 3001 rvnfaqtell ihvvnrdtnr ildvdrviqm idenkeqlrn lfrnynvldv qpaisvrlpd 3061 dmsalqmaii vlaillflaa mlfvlmnwyy rtvhkrklka ivagsagnrg fidimdmpnt 3121 nkysfdganp vwldpfcrnl elaaqaehed dlpenlseia dlwnsptrth gtfgrepaav 3181 kpdddrylra aiqeydniak lgqiiregpi kgsllkvvle dylrlkklfa qrmvqkassc 3241 hssiseliqt eldeepgdhs pgqgslrfrh kppvelkgpd gihvvhgstg tllatdlnsl 3301 peedqkglgr sletltaaea tafernarte sakstplhkl rdvimetple itel';

protein();

}

} else if (randomizer <= 45) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = '1 magwpgagpl cvlggaalgv clagvagqlv epstappkpk pppltketvv fwdmrlwhvv 61 gifslfvlsi iitlccvfnc rvprtrkeie arylqrkaak mytdkletvp plneltevpg 121 edkkkkkkkk kdsvdtvaik veedekneak kkkgek';

} else if (randomizer <= 40) {

peptidesequence = '1 mspkkvqikv eekedetees sseeeeeved klprreslrp krkrtrdvin eddpepeped 61 eetrkareke rrrrlkrgae eeeideeele rlkaeldekr qiiatvkckp wkmekkievl 121 keakkfvsen egalgkgkgk rwfafkmmma kkwakflrdf enfkaacvpw enkikaiesq 181 fgssvasyfl flrwmygvnm vlfiltfsli mlpeylwglp ygslprktvp raeeasaanf 241 gvlydfngla qysvlfygyy dnkrtigwmn frlplsyflv gimcigysfl vvlkamtkni 301 gddgggddnt fnfswkvfts wdylignpet adnkfnsitm nfkeaiteek aaqveenvhl 361 irflrflanf fvfltlggsg ylifwavkrs qefaqqdpdt lgwweknemn mvmsllgmfc 421 ptlfdlfael edyhplialk wllgrifall lgnlyvfila lmdeinnkie eeklvkanit 481 lweanmikay nasfsenstg ppffvhpadv prgpcwetmv gqefvrltvs dvlttyvtil 541 igdflracfv rfcnycwcwd leygypsyte fdisgnvlal ifnqgmiwmg sffapslpgi 601 nilrlhtsmy fqcwavmccn vpearvfkas rsnnfylgml llilflstmp vlymivslpp 661 sfdcgpfsgk nrmfeviget lehdfpswma kilrqlsnpg lviavilvmv laiyylnata 721 kgqkaanldl kkkmkmqale nkmrnkkmaa araaaaagrq';

} else {

peptidesequence = ' 1 mvkllpaqea akiyhtnyvr nsravgvmwg tlticfsvlv malfiqpywi gdsvntpqag 61 yfglfsycvg nvlsselick ggpldfssip srafktamff valgmfliig siicfslffi 121 cntatvykic awmqlaaatg lmigclvypd gwdssevrrm cgeqtgkytl ghctirwafm 181 lailsigdal ilsflafvlg yrqdkllpdd ykadgteev';

}

protein();

} else {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = '1 mrecisihvg qagvqignac welyclehgi qpdgqmpsdk tigggddsfn tffsetgagk 61 hvpravfvdl eptvidevrt gtyrqlfhpe qlitgkedaa nnyarghyti gkeiidlvld 121 rirkladqct glqgflvfhs fgggtgsgft sllmerlsvd ygkksklefs iypapqvsta 181 vvepynsilt thttlehsdc afmvdneaiy dicrrnldie rptytnlnrl igqivssita 241 slrfdgalnv dltefqtnlv pyprihfpla tyapvisaek ayheqlsvae itnacfepan 301 qmvkcdprhg kymaccllyr gdvvpkdvna aiatiktkrt iqfvdwcptg fkvginyqpp 361 tvvpggdlak vqravcmlsn ttaiaeawar ldhkfdlmya krafvhwyvg egmeegefse 421 aredmaalek dyeevgvdsv egegeeegee y';

} else if (randomizer <= 40) {

peptidesequence = '1 mgdvgprwan wdpsrsgvrd hpgqqsetlp hatflqkikf wevisdehgi dptgtyhgds 61 dlqldrisvy yneatggkyv prailvdlep gtmdsvrsgp fgqifrpdnf vfgqsgagnn 121 wakghytega elvdsvldvv rkeaescdcl qgfqlthslg ggtgsgmgtl liskireeyp 181 drimntfsvv pspkvsdtvv epynatlsvh qlventdety cidnealydi cfrtlklttp 241 tygdlnhlvs atmsgvttcl rfpgqlnadl rklavnmvpf prlhffmpgf apltsrgsqq 301 yraltvpelt qqvfdaknmm aacdprhgry ltvaavfrgr msmkevdeqm lnvqnknssy 361 fvewipnnvk tavcdipprg lkmavtfign staiqelfkr iseqftamfr rkaflhwytg 421 egmdemefte aesnmndlvs eyqqyqdata eeeedfgeea eeea';

} else {

peptidesequence = '1 mpreiitlql gqcgnqigfe fwkqlcaehg ispegiveef ategtdrkdv ffyqaddehy 61 ipravlldle prvihsilns pyaklynpen iylsehggga gnnwasgfsq gekihedifd 121 iidreadgsd slegfvlchs iaggtgsglg syllerlndr ypkklvqtys vfpnqdemsd 181 vvvqpynsll tlkrltqnad cvvvldntal nriatdrlhi qnpsfsqinq lvstimsast 241 ttlrypgymn ndligliasl iptprlhflm tgytplttdq svasvrkttv ldvmrrllqp 301 knvmvstgrd rqtnhcyiai lniiqgevdp tqvhkslqri rerklanfip wgpasiqval 361 srkspylpsa hrvsglmman htsisslfer tcrqydklrk reafleqfrk edmfkdnfde 421 mdtsreivqq lideyhaatr pdyiswgtqe q';

}

protein();

}

} else {

if (randomizer <= 20) {

peptidesequence = '1 mcdedettal vcdngsglvk agfagddapr avfpsivgrp rhqgvmvgmg qkdsyvgdea 61 qskrgiltlk ypiehgiitn wddmekiwhh tfynelrvap eehptlltea plnpkanrek 121 mtqimfetfn vpamyvaiqa vlslyasgrt tgivldsgdg vthnvpiyeg yalphaimrl 181 dlagrdltdy lmkiltergy sfvttaerei vrdikeklcy valdfenema taasssslek 241 syelpdgqvi tignerfrcp etlfqpsfig mesagihett ynsimkcdid irkdlyannv 301 msggttmypg iadrmqkeit alapstmkik iiapperkys vwiggsilas lstfqqmwit 361 kqeydeagps ivhrkcf';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 30) {

peptidesequence = '1 mfrqfylwtc lasgiilgsl feiclgqydd dwqyedckla rggppativa ideesrngti 61 lvdnmlikgt aggpdptiel slkdnvdywv lmdpvkqmlf lnstgrvldr dppmnihsiv 121 vqvqcinkkv gtiiyhevri vvrdrndnsp tfkhesyyat vneltpvgtt iftgfsgdng 181 atdiddgpng qieyviqynp ddptsndtfe iplmltgniv lrkrlnyedk tryfviiqan 241 draqnlnerr tttttltvdv ldgddlgpmf lpcvlvpntr dcrpltyqaa ipelrtpeel 301 npiivtppiq aidqdrniqp psdrpgilys ilvgtpedyp rffhmhprta elsllepvnr 361 dfhqkfdlvi kaeqdnghpl pafaglhiei ldennqspyf tmpsyqgyil esapvgatis 421 dslnltsplr ivaldkdied tkdpelhlfl ndytsvftvt qtgitryltl lqpvdreeqq 481 tytfsitafd gvqesepviv niqvmdandn tptfpeisyd vyvytdmrpg dsviqltavd 541 adegsngeit yeilvgaqgd fiinkttgli tiapgvemiv grtyaltvqa adnappaerr 601 nsictvyiev lppnnqsppr fpqlmyslei seamrvgavl lnlqatdreg dsityaieng 661 dpqrvfnlse ttgiltlgka ldrestdryi liitasdgrp dgtstatvni vvtdvndnap 721 vfdpylprnl svveeeanaf vgqvkatdpd agingqvhys lgnfnnlfri tsngsiytav 781 klnrevrdyy elvvvatdga vhprhstltl aikvldiddn spvftnstyt vlveenlpag 841 ttilqieakd vdlganvsyr irspevkhff alhpftgels llrsldyeaf pdqeasitfl 901 veafdiygtm ppgiatvtvi vkdmndyppv fskriykgmv apdavkgtpi ttvyaedadp 961 pglpasrvry rvddvqfpyp asifeveeds grvitrvnln eepttifklv vvafddgepv 1021 msssatvkil vlhpgeiprf tqeeyrpppv selatkgtmv gvisaaainq sivysivsgn 1081 eedtfginni tgviyvngpl dyetrtsyvl rvqadslevv lanlrvpsks ntakvyieiq 1141 dennhppvfq kkfyiggvse darmftsvlr vkatdkdtgn ysvmayrlii ppikegkegf 1201 vvetytglik tamlfhnmrr syfkfqviat ddygkglsgk advlvsvvnq ldmqvivsnv 1261 pptlvekkie dlteildryv qeqipgakvv vesigarrhg dafsledytk cdltvyaidp 1321 qtnraidrne lfkfldgkll dinkdfqpyy geggrileir tpeavtsikk rgeslgyteg 1381 allalafiii lccipailvv lvsyrqfkvr qaectktari qaalpaakpa vpapapvaap 1441 ppppppppga hlyeelgdss mhnlfllyhf qqsrgnnsvs edrkhqqvvm pfssntieah 1501 ksahvdgslk snklksarkf tflsdeddls ahnplykeni sqvstnsdis qrtdfvdpfs 1561 pkiqaksksl rgprekiqrl wsqsvslprr lmrkvpnrpe iidlqqwqgt rqkaenentg 1621 ictnkrgssn plltteeanl tekeeirqge tlmiegteql kslssdssfc fprphfsfst 1681 lptvsrtvel ksepnvissp aecslelsps rpcvlhssls rretpicmlp ieternifen 1741 fahppnisps acplpppppi sppspppapa plapppdisp fslfcpppsp psiplplppp 1801 tffplsvsts gpptppllpp fptplppppp sipcppppsa sflstecvci tgvkcttnlm 1861 paekikssmt qlstttvckt dpqrepkgil rhvknlaele ksvanmysqi eknylrtnvs 1921 elqtmcpsev tnmeitseqn kgslnniveg tekqshsqst sl';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgrhvatsch vawllvlisg cwgqvnrlpf ftnhffdtyl lisedtpvgs svtqllaqdm 61 dndplvfgvs geeasrffav epdtgvvwlr qpldretkse ftvefsvsdh qgvitrkvni 121 qvgdvndnap tfhnqpysvr ipentpvgtp ifivnatdpd lgaggsvlys fqppsqffai 181 dsargivtvi reldyettqa yqltvnatdq dktrplstla nlaiiitdvq dmdpifinlp 241 ystniyehsp pgttvriita idqdkgrprg igytivsgnt nsifaldyis gvltlnglld 301 renplyshgf iltvkgteln ddrtpsdatv tttfnilvid indnapefns seysvaitel 361 aqvgfalplf iqvvdkdenl glnsmfevyl vgnnshhfii sptsvqgkad irirvaipld 421 yetvdrydfd lfanesvpdh vgyakvkitl inendnrpif sqplynisly envtvgtsvl 481 tvlatdndag tfgevsyffs ddpdrfsldk dtglimliar ldyeliqrft ltiiardggg 541 eettgrvrin vldvndnvpt fqkdayvgal renepsvtql vrlratdeds ppnnqitysi 601 vsasafgsyf dislyegygv isvsrpldye qisngliylt vmamdagnpp lnstvpvtie 661 vfdendnppt fskpayfvsv venimagatv lflnatdldr sreygqesii yslegstqfr 721 inarsgeitt tslldretks eyilivravd ggvghnqktg iatvnitlld indnhptwkd 781 apyyinlvem tppdsdvttv vavdpdlgen gtlvysiqpp nkfyslnstt gkirtthaml 841 drenpdphea elmrkivvsv tdcgrpplka tssatvfvnl ldlndndptf qnlpfvaevl 901 egipagvsiy qvvaidldeg lnglvsyrmp vgmprmdfli nsssgvvvtt teldreriae 961 yqlrvvasda gtptksstst ltihvldvnd etptffpavy nvsvsedvpr efrvvwlnct 1021 dndvglnael syfitggnvd gkfsvgyrda vvrtvvgldr ettaaymlil eaidngpvgk 1081 rhtgtatvfv tvldvndnrp iflqssyeas vpedipeghs ilqlkatdad egefgrvwyr 1141 ilhgnhgnnf rihvsngllm rgprpldrer nsshvlivea ynhdlgpmrs svrvivyved 1201 indeapvftq qqysrlglre tagigtsviv vqatdrdsgd gglvnyrils gaegkfeide 1261 stgliitvny ldyetktsym mnvsatdqap pfnqgfcsvy itllneldea vqfsnasyea 1321 ailenlalgt eivrvqaysi dnlnqityrf naytstqaka lfkidaitgv itvqglvdre 1381 kgdfytltvv addggpkvds tvkvyitvld endnsprfdf tsdsavsipe dcpvgqrvat 1441 vkawdpdags ngqvvfslas gniagafeiv ttndsigevf varpldreel dhyilqvvas 1501 drgtpprkkd hilqvtildi ndnppviesp fgynvsvnen vgggtavvqv ratdrdigin 1561 svlsyyiteg nkdmafrmdr isgeiatrpa ppdrerqsfy hlvatvedeg tptlsatthv 1621 yvtivdendn apmfqqphye vlldegpdtl ntslitiqal dldegpngtv tyaivagniv 1681 ntfridrhmg vitaakeldy eishgrytli vtatdqcpil shrltstttv lvnvndindn 1741 vptfprdyeg pfevtegqpg prvwtflahd rdsgpngqve ysimdgdplg efvispvegv 1801 lrvrkdveld retiafynlt icardrgmpp lsstmlvgir vldindndpv llnlpmniti 1861 senspvssfv ahvlasdads gcnarltfni tagnreraff inattgivtv nrpldrerip 1921 eykltisvkd npenpriarr dydlllifls dendnhplft kstyqaevme nspagtpltv 1981 lngpilalda dqdiyavvty qllgaqsglf dinsstgvvt vrsgviidre afsppilell 2041 llaediglln stahllitil ddndnrptfs patltvhlle ncppgfsvlq vtatdedsgl 2101 ngelvyriea gaqdrflihl vtgvirvgna tidreeqesy rltvvatdrg tvplsgtaiv 2161 tiliddinds rpeflnpiqt vsvlesaepg tvianitaid hdlnpkleyh ivgivakddt 2221 drlvpnqeda favnintgsv mvkspmnrel vatyevtlsv idnasdlper svsvpnaklt 2281 vnvldvndnt pqfkpfgity ymerilegat pgttliavaa vdpdkglngl vtytlldlvp 2341 pgyvqledss agkvianrtv dyeevhwlnf tvrasdngsp praaeipvyl eivdindnnp 2401 ifdqpsyqea vfedvpvgti iltvtatdad sgnfalieys lgdgeskfai npttgdiyvl 2461 ssldrekkdh yiltalakdn pgdvasnrre nsvqvviqvl dvndcrpqfs kpqfstsvye 2521 nepagtsvit mmatdqdegp ngeltysleg pgveafhvdm dsglvttqrp lqsyekfslt 2581 vvatdggepp lwgttmllve vidvndnrpv fvrppngtil hireeiplrs nvyevyatdk 2641 deglngavry sflktagnrd weffiidpis gliqtaqrld resqavysli lvasdlgqpv 2701 pyetmqplqv alediddnep lfvrppkgsp qyqlltvpeh sprgtlvgnv tgavdadegp 2761 naivyyfiaa gneeknfhlq pdgcllvlrd ldrereaifs fivkassnrs wtpprgpspt 2821 ldlvadltlq evrvvledin dqpprftkae ytagvatdak vgseliqvla ldadignnsl 2881 vfysilaihy fralandsed vgqvftmgsm dgilrtfdlf mayspgyfvv divardlagh 2941 ndtaiigiyi lrddqrvkiv ineipdrvrg feeefihlls nitgaivntd nvqfhvdkkg 3001 rvnfaqtell ihvvnrdtnr ildvdrviqm idenkeqlrn lfrnynvldv qpaisvrlpd 3061 dmsalqmaii vlaillflaa mlfvlmnwyy rtvhkrklka ivagsagnrg fidimdmpnt 3121 nkysfdganp vwldpfcrnl elaaqaehed dlpenlseia dlwnsptrth gtfgrepaav 3181 kpdddrylra aiqeydniak lgqiiregpi kgsllkvvle dylrlkklfa qrmvqkassc 3241 hssiseliqt eldeepgdhs pgqgslrfrh kppvelkgpd gihvvhgstg tllatdlnsl 3301 peedqkglgr sletltaaea tafernarte sakstplhkl rdvimetple itel';

protein();

}

} else if (randomizer <= 60) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = '1 magwpgagpl cvlggaalgv clagvagqlv epstappkpk pppltketvv fwdmrlwhvv 61 gifslfvlsi iitlccvfnc rvprtrkeie arylqrkaak mytdkletvp plneltevpg 121 edkkkkkkkk kdsvdtvaik veedekneak kkkgek';

} else if (randomizer <= 40) {

peptidesequence = '1 mspkkvqikv eekedetees sseeeeeved klprreslrp krkrtrdvin eddpepeped 61 eetrkareke rrrrlkrgae eeeideeele rlkaeldekr qiiatvkckp wkmekkievl 121 keakkfvsen egalgkgkgk rwfafkmmma kkwakflrdf enfkaacvpw enkikaiesq 181 fgssvasyfl flrwmygvnm vlfiltfsli mlpeylwglp ygslprktvp raeeasaanf 241 gvlydfngla qysvlfygyy dnkrtigwmn frlplsyflv gimcigysfl vvlkamtkni 301 gddgggddnt fnfswkvfts wdylignpet adnkfnsitm nfkeaiteek aaqveenvhl 361 irflrflanf fvfltlggsg ylifwavkrs qefaqqdpdt lgwweknemn mvmsllgmfc 421 ptlfdlfael edyhplialk wllgrifall lgnlyvfila lmdeinnkie eeklvkanit 481 lweanmikay nasfsenstg ppffvhpadv prgpcwetmv gqefvrltvs dvlttyvtil 541 igdflracfv rfcnycwcwd leygypsyte fdisgnvlal ifnqgmiwmg sffapslpgi 601 nilrlhtsmy fqcwavmccn vpearvfkas rsnnfylgml llilflstmp vlymivslpp 661 sfdcgpfsgk nrmfeviget lehdfpswma kilrqlsnpg lviavilvmv laiyylnata 721 kgqkaanldl kkkmkmqale nkmrnkkmaa araaaaagrq';

} else {

peptidesequence = ' 1 mvkllpaqea akiyhtnyvr nsravgvmwg tlticfsvlv malfiqpywi gdsvntpqag 61 yfglfsycvg nvlsselick ggpldfssip srafktamff valgmfliig siicfslffi 121 cntatvykic awmqlaaatg lmigclvypd gwdssevrrm cgeqtgkytl ghctirwafm 181 lailsigdal ilsflafvlg yrqdkllpdd ykadgteev';

}

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

cursor2x = cursor2x - 1 \* 1.23758547e+29;

}

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2y = cursor2y - 1 \* 1.23758547e+29;

earb = earb - 3;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.01 \* 3.093963676e+28;

cursor1z = spawnerz + 10 \* 3.093963676e+28;

repeat\_endcell1 = Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(6.187927353e+29 / 4.950341882e+27 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

synaptic\_vesicle\_1();

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

cursor1z = cursor1z + 4.950341882e+27;

}

cursor1z = cursor1z - 4.950341882e+27 \* Math.round(6.187927353e+29 / 4.950341882e+27 - 2);

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.01 \* 3.093963676e+28;

cursor1z = spawnerz + 1 \* 1.546981838e+30;

repeat\_endcell1 = Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(6.187927353e+29 / 6.806720089e+28 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

mitochondrion();

cursor1y = cursor1y - 3.712756412e+28;

}

cursor1y = cursor1y + 3.712756412e+28 \* Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

cursor1z = cursor1z + 6.806720089e+28;

}

cursor1z = cursor1y - 6.806720089e+28 \* Math.round(6.187927353e+29 / 6.806720089e+28 - 2);

cursor1x = cursor1x - 3.712756412e+28;

}

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 2 \* 3.093963676e+28;

cursor1z = spawnerz + 9.28189103e+29 \* 1;

nucleoid\_size\_y = 4.331549147e+29;

nucleoid\_size\_z = 4.331549147e+29;

nucleoid();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + 9.28189103e+29 \* 1;

vital\_components();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + (2.134834937e+29 + 9.28189103e+29 \* 1);

vital\_components2();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + (4.331549147e+29 + 9.28189103e+29 \* 1);

vital\_components3();

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.1 \* 3.093963676e+28;

cursor1z = spawnerz + 9.28189103e+29 \* 1;

for (countcell1 = 0; countcell1 < 4300; countcell1++) {

set\_earth\_rotation();

cursor1x = cursor1x - 1 \* 1.23758547e+29;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mprsflvksk kahsyhqprs pgpdyslrle nvpapsrads tsnaggakae prdrlspesq 61 lteapdrasa spdscegsvc erssefedfw rppspsaspa seksmcpsld eaqpfplpfk 121 pyswsglags dlrhlvqsyr pcgalergag lglfcepape pghpaalygp kraaggagag 181 apgscsagag atagpglgly gdfgsaaagl yerptaaagl lyperghglh adkgagvkve 241 sellctrlll gggsykcikc skvfstphgl evhvrrshsg trpfacemcg ktfghavsle 301 qhkavhsqer sfdckicgks fkrsstlsth llihsdtrpy pcqycgkrfh qksdmkkhtf 361 ihtgekphkc qvcgkafsqs snlithsrkh tgfkpfgcdl cgkgfqrkvd lrrhretqhg 421 lk';

} else if (randomizer <= 60) {

peptidesequence = '1 mmamnskqpf gmhpvlqepk fsslhsgsea mrrvclpapq lqgnifgsfd esllaraeal 61 aavdivshgk nhpfkpdaty htmssvpcts tsstvpishp aaltshphha vhqglegdll 121 ehisptlsvs glgapehsvm paqihphhlg amghlhqamg mshphtvaph sampaclsdv 181 esdpreleaf aerfkqrrik lgvtqadvga alanlkipgv gslsqsticr fesltlshnn 241 mialkpvlqa wleeaeaayr eknskpelfn gserkrkrts iaapekrsle ayfaiqprps 301 sekiaaiaek ldlkknvvrv wfcnqrqkqk rmkysavh';

} else {

peptidesequence = '1 msrllhaeew aevkelgdhh rqpqphhlpq pppppqppat lqarehpvyp pelslldstd 61 prawlaptlq gictaraaqy llhspelgas eaaaprdevd grgelvrrss ggassskspg 121 pvkvreqlck lkggvvvdel gcsrqrapss kqvngvqkqr rlaanarerr rmhglnhafd 181 qlrnvipsfn ndkklskyet lqmaqiyina lsellqtpsg geqpppppas cksdhhhlrt 241 aasyeggagn ataagaqqas ggsqrptppg scrtrfsapa saggysvqld alhfstfeds 301 altammaqkn lspslpgsil qpvqeenskt sprshrsdge fsphshysds deas';

}

protein();

}

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2z = spawnerz + 9 \* 3.093963676e+28;

lipid1 = 5.569134618e+29;

lipid2 = lipid1;

lipid\_wall\_z();

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2z = spawnerz + 9 \* 3.093963676e+28;

lipid2 = lipid1;

cursor2z = spawnerz + (9 \* 3.093963676e+28 - 0.1 \* 3.093963676e+28);

lipid\_wall\_z();

cursor2z = spawnerz + 1 \* 3.093963676e+28;

lipid2 = 7.9 \* 3.093963676e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid2 = 1 \* 5.569134618e+29;

lipid1 = 2.165774573e+29;

lipid\_wall\_z();

cursor2x = spawnerx - 1 \* 3.712756412e+29;

lipid\_wall\_z();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid2 = 2.165774573e+29;

lipid1 = 1 \* 5.569134618e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.712756412e+29;

lipid\_wall\_y();

cursor2z = spawnerz + 1 \* 1;

cursor2y = spawnery - 1 \* 2.475170941e+29;

cursor2x = spawnerx - 1 \* 2.475170941e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 1 \* 1.23758547e+29;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.712756412e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 2.475170941e+29;

cursor2x = spawnerx - 1 \* 3.712756412e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1 \* 3.093963676e+28;

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1z = spawnerz + (9 \* 3.093963676e+28 - 0.05 \* 3.093963676e+28);

for (countcell1 = 0; countcell1 < 90; countcell1++) {

for (countcell2 = 0; countcell2 < 90; countcell2++) {

synapse\_z();

cursor1y = cursor1y - 6.187927353e+30;

}

cursor1y = cursor1y + 6.187927353e+30 \* 90;

cursor1x = cursor1x - 6.187927353e+30;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.05 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 2.072955663e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 2.072955663e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.15 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 2.072955663e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.2 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 2.072955663e+30) / density\_parameter\_z - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((20 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 1.856378206e+30) / density\_parameter\_z - 8);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((20 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 1.856378206e+30) / density\_parameter\_z - 8);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 5.569134618e+29;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((20 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 1.856378206e+30) / density\_parameter\_z - 8);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 5.569134618e+29;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((20 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 1.856378206e+30) / density\_parameter\_z - 8);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round((2.784567309e+30 + eara) / density\_parameter\_x - 0);

}

for (countsub1 = 0; countsub1 < generation\_parameter\_3; countsub1++) {

for (countsub2 = 0; countsub2 < generation\_parameter\_x; countsub2++) {

for (countsub3 = 0; countsub3 < generation\_parameter\_z; countsub3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_cytosol();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

list\_cells.push(['audiocyte', spawnerz, spawnery, spawnerx, spawnerz + (2.784567309e+30 + eara), spawnery - 6.187927352e+29, spawnerx - 6.187927352e+29]);

}

}

function rod\_cell() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.187927352e+30, spawnery - 6.187927352e+29, spawnerx - 6.187927352e+29];

make = true;

exempt = [null];

screen\_molecules();

screen\_cells();

if (make == true) {

lipid2 = 5.847591349e+30;

lipid1 = 5.569134618e+29;

cursor2z = spawnerz + 10 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

lipid2 = lipid1;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2z = spawnerz + 1 \* 6.156987716e+30;

lipid\_wall\_z();

if (unused\_variable == 5) {

cursor2x = spawnerx - 1 \* 6.187927353e+28;

cursor2y = spawnery - 1 \* 6.187927353e+28;

cursor2z = spawnerz + 1 \* 3.093963676e+30;

lipid1 = 4.950341882e+29;

lipid2 = lipid1;

repeat\_endcell1 = Math.round(10 \* 200 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

lipid\_wall\_z();

cursor1z = cursor1z + 1 \* 1.546981838e+27;

}

if (unused\_variable == 5) {

cursor2x = spawnerx - 1 \* 6.187927353e+28;

cursor2y = spawnery - 1 \* 6.187927353e+28;

cursor2z = spawnerz + 1 \* 3.093963676e+30;

lipid1 = 1.546981838e+27;

lipid2 = 4.950341882e+29;

repeat\_endcell1 = Math.round(10 \* 100 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

lipid\_wall\_y();

cursor1z = cursor1z + 1 \* 1.546981838e+27;

}

}

if (unused\_variable == 5) {

cursor2x = spawnerx - 1 \* 6.187927353e+28;

cursor2y = spawnery - 1 \* 6.187927353e+28;

cursor2z = spawnerz + 1 \* 3.093963676e+30;

lipid1 = 4.950341882e+29;

lipid2 = 1.546981838e+27;

repeat\_endcell1 = Math.round(10 \* 100 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

lipid\_wall\_x();

cursor1z = cursor1z + 1 \* 1.546981838e+27;

}

}

if (unused\_variable == 5) {

cursor2x = spawnerx - 1 \* 6.187927353e+28;

cursor2y = spawnery - 1 \* 5.569134618e+29;

cursor2z = spawnerz + 1 \* 3.093963676e+30;

lipid1 = 1.546981838e+27;

lipid2 = 4.950341882e+29;

repeat\_endcell1 = Math.round(10 \* 100 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

lipid\_wall\_y();

cursor1z = cursor1z + 1 \* 1.546981838e+27;

}

}

if (unused\_variable == 5) {

cursor2x = spawnerx - 1 \* 5.569134618e+29;

cursor2y = spawnery - 1 \* 6.187927353e+28;

cursor2z = spawnerz + 1 \* 3.093963676e+30;

lipid1 = 4.950341882e+29;

lipid2 = 1.546981838e+27;

repeat\_endcell1 = Math.round(10 \* 100 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

lipid\_wall\_x();

cursor1z = cursor1z + 1 \* 1.546981838e+27;

}

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.01 \* 3.093963676e+28;

cursor1z = spawnerz + 10 \* 3.093963676e+28;

repeat\_endcell1 = Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(1.23758547e+30 / 4.950341882e+27 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

synaptic\_vesicle\_1();

cursor1y = cursor1y - 4.950341882e+27;

}

cursor1y = cursor1y + 4.950341882e+27 \* Math.round(5.569134618e+29 / 4.950341882e+27 - 2);

cursor1z = cursor1z + 4.950341882e+27;

}

cursor1z = cursor1z - 4.950341882e+27 \* Math.round(1.23758547e+30 / 4.950341882e+27 - 2);

cursor1x = cursor1x - 4.950341882e+27;

}

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.01 \* 3.093963676e+28;

cursor1z = spawnerz + 10 \* 2.165774573e+29;

repeat\_endcell1 = Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

for (countcell1 = 0; countcell1 < repeat\_endcell1; countcell1++) {

repeat\_endcell2 = Math.round(9.28189103e+29 / 6.806720089e+28 - 2);

for (countcell2 = 0; countcell2 < repeat\_endcell2; countcell2++) {

repeat\_endcell3 = Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

for (countcell3 = 0; countcell3 < repeat\_endcell3; countcell3++) {

mitochondrion();

cursor1y = cursor1y - 3.712756412e+28;

}

cursor1y = cursor1y + 3.712756412e+28 \* Math.round(5.569134618e+29 / 3.712756412e+28 - 2);

cursor1z = cursor1z + 6.806720089e+28;

}

cursor1z = cursor1z - 6.806720089e+28 \* Math.round(9.28189103e+29 / 6.806720089e+28 - 2);

cursor1x = cursor1x - 3.712756412e+28;

}

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 2 \* 3.093963676e+28;

cursor1z = spawnerz + 1.553169765e+30 \* 1;

nucleoid\_size\_y = 4.331549147e+29;

nucleoid\_size\_z = 4.331549147e+29;

nucleoid();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + 1.553169765e+30 \* 1;

vital\_components();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + (2.134834937e+29 + 1.553169765e+30 \* 1);

vital\_components2();

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 16 \* 3.093963676e+28;

cursor1z = spawnerz + (4.331549147e+29 + 1.553169765e+30 \* 1);

vital\_components3();

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

cursor1y = spawnery - 1.1 \* 3.093963676e+28;

cursor1z = spawnerz + 1.553169765e+30 \* 1;

for (countcell1 = 0; countcell1 < 4300; countcell1++) {

set\_earth\_rotation();

cursor1x = cursor1x - 1 \* 1.23758547e+29;

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mmaymnpgph ysvnalalsg psvdlmhqav pypsaprkqr rerttftrsq leelealfak 61 tqypdvyare evalkinlpe srvqvwfknr rakcrqqrqq qkqqqqppgg qakarpakrk 121 agtsprpstd vcpdplgisd syspplpgps gspttavatv siwspasesp lpeaqraglv 181 asgpsltsap yamtyapasa fcsspsaygs pssyfsgldp ylspmvpqlg gpalsplsgp 241 svgpslaqsp tslsgqsyga yspvdslefk dptgtwkfty npmdpldykd qsawkfqil';

} else if (randomizer <= 60) {

peptidesequence = ' 1 mhlpgcapam adgsfslagh llrspggsts rlhsieailg ftkddgilgt fpaergarga 61 kerdrrlgar pacpkapeeg sepspppapa papeyeaprp ycpkepgear pspglpvgpa 121 tgeaklseee qpkkkhrrnr ttfttyqlhe lerafekshy pdvysreela gkvnlpevrv 181 qvwfqnrrak wrrqeklevs smklqdspll sfsrsppsat lsplgagpgs gggpaggalp 241 leswlgpplp gggatalqsl pgfgppaqsl pasytppppp ppflnspplg pglqplappp 301 psypcgpgfg dkfpldeadp rnssiaalrl kakehiqaig kpwqal';

} else {

peptidesequence = '1 mtksysesgl mgepqpqgpp swtdeclssq deeheadkke ddletmnaee dslrnggeee 61 dededleeee eeeeedddqk pkrrgpkkkk mtkarlerfk lrrmkanare rnrmhglnaa 121 ldnlrkvvpc ysktqklski etlrlaknyi walseilrsg kspdlvsfvq tlckglsqpt 181 tnlvagclql nprtflpeqn qdmpphlpta sasfpvhpys yqspglpspp ygtmdsshvf 241 hvkppphays aalepffesp ltdctspsfd gplspplsin gnfsfkheps aefeknyaft 301 mhypaatlag aqshgsifsg taaprceipi dnimsfdshs hhervmsaql naifhd';

}

protein();

}

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2z = spawnerz + 9 \* 3.093963676e+28;

lipid1 = 5.569134618e+29;

lipid2 = lipid1;

lipid\_wall\_z();

lipid2 = 1 \* 3.093963676e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2z = spawnerz + 9 \* 3.093963676e+28;

lipid2 = lipid1;

cursor2z = spawnerz + (9 \* 3.093963676e+28 - 0.1 \* 3.093963676e+28);

lipid\_wall\_z();

cursor2z = spawnerz + 1 \* 3.093963676e+28;

lipid2 = 7.9 \* 3.093963676e+28;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 5.878530986e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.093963676e+28;

cursor2x = spawnerx - 1 \* 5.878530986e+29;

lipid\_wall\_x();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid2 = 1 \* 5.569134618e+29;

lipid1 = 2.165774573e+29;

lipid\_wall\_z();

cursor2x = spawnerx - 1 \* 3.712756412e+29;

lipid\_wall\_z();

cursor2x = spawnerx - 1 \* 3.093963676e+28;

lipid2 = 2.165774573e+29;

lipid1 = 1 \* 5.569134618e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.712756412e+29;

lipid\_wall\_y();

cursor2z = spawnerz + 1 \* 1;

cursor2y = spawnery - 1 \* 2.475170941e+29;

cursor2x = spawnerx - 1 \* 2.475170941e+29;

lipid2 = 1 \* 3.093963676e+28;

lipid1 = 1 \* 1.23758547e+29;

lipid\_wall\_x();

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 3.712756412e+29;

lipid\_wall\_y();

cursor2y = spawnery - 1 \* 2.475170941e+29;

cursor2x = spawnerx - 1 \* 3.712756412e+29;

lipid\_wall\_x();

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 1 \* 3.093963676e+28;

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1z = spawnerz + (9 \* 3.093963676e+28 - 0.05 \* 3.093963676e+28);

for (countcell1 = 0; countcell1 < 90; countcell1++) {

for (countcell2 = 0; countcell2 < 90; countcell2++) {

synapse\_z();

cursor1y = cursor1y - 6.187927353e+30;

}

cursor1y = cursor1y + 6.187927353e+30 \* 90;

cursor1x = cursor1x - 6.187927353e+30;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 15) {

peptidesequence = '1 msrrkisses fsslgsdyle tspeeegecp lsrlcwngsr sppgplepsp aaaaaaaapa 61 ptpaasaaaa aatagarrvq rrrrvnldsl gesisrltap spqtiqqtlk rtlqyyehqv 121 igyrdaeknf hnisnrcsya dhsnkeeied vsgilqctan ilglkfeeiq krfgeeffni 181 cfhenervlr avggtlqdff ngfdallehi rtsfgkqatl espsflckel pegtlmlhyf 241 hphhivgfam lgmikaagkk iyrldveveq vaneklcsdv snpgncsclt flikecentn 301 imknlpqgts qvpadlrisi ntfcrafpfh lmfdpsmsvl qlgeglrkql rcdthkvlkf 361 edcfeivspk vnatfervll rlstpfvirt kpeasgsenk dkvmevkgqm ihvpesnsil 421 flgspcvdkl delmgrglhl sdipihdatr dvilvgeqak aqdglkkrmd klkatlerth 481 qaleeekkkt vdllysifpg dvaqqlwqgq qvqarkfddv tmlfsdivgf taicaqctpm 541 qvismlnely trfdhqcgfl diykvetigd aycvaaglhr kslchakpia lmalkmmels 601 eevltpdgrp iqpqrsellf sfpvsiqlvp dqhqsetdlg tekmrigihs gsvlagvvgv 661 rmpryclfgn nvtlaskfes gshprrinvs pttyqllkre esftfiprsr eelpdnfpke 721 ipgicyflev rtgpkppkps lsssrikkvs ynigtmflre tsl';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mfctklkdlk itgecpfsll apgqvpness eeaagssesc katvpicqdi pekniqeslp 61 qrktsrsrvy lhtlaesick lifpeferln valqrtlakh kikesrksle redfektiae 121 qavaagvpve vikeslgeev fkicyeeden ilgvvggtlk dflnsfstll kqsshcqeag 181 krgrledasi lcldkeddfl hvyyffpkrt tslilpgiik aaahvlyete vevslmppcf 241 hndcsefvnq pyllysvhmk stkpslspsk pqsslvipts lfcktfpfhf mfdkdmtilq 301 fgngirrlmn rrdfqgkpnf eeyfeiltpk inqtfsgimt mlnmqfvvrv rrwdnsvkks 361 srvmdlkgqm iyivessail flgspcvdrl edftgrglyl sdipihnalr dvvligeqar 421 aqdglkkrlg klkatleqah qaleeekkkt vdllcsifpc evaqqlwqgq vvqakkfsnv 481 tmlfsdivgf taicsqcspl qvitmlnaly trfdqqcgel dvykvetigd aycvagglhk 541 esdthavqia lmalkmmels devmsphgep ikmriglhsg svfagvvgvk mpryclfgnn 601 vtlankfesc svprkinvsp ttyrllkdcp gfvftprsre elppnfpsei pgichflday 661 qqgtnskpcf qkkdvedgna nflgkasgid';

protein();

} else if (randomizer <= 45) {

peptidesequence = '1 msgydrmlrt lggnlmefie nldalhsyla lsyqemnaps frvergadgk mflhyysdrs 61 glchivpgii eavakdffdi dvimdildmn eevertgkke hvvflivqka hrkmrktkpk 121 rlqdsqgmer dqealqaafl kmkekylnvs acpvkkshwd vvrsivmfgk ghlmntfepi 181 yperlwieek tfcnafpfhi vfdeslqvkq arvniqkyvp glqtqniqld eyfsiihpqv 241 tfnifsirrf insqfvlktr remmpvawqs rttlklqgqm iwmesmwcmv ylcspklrsl 301 qeleelnmhl sdiapndttr dlillnqqrl aeielsnqle rkkeelqvls khlaiekkkt 361 etllyamlpk hvanqlregk kvaagefksc tilfsdvvtf tnictacepi qivnvlnsmy 421 skfdrltsvh avykvetigd aymvvggvpv pignhaqrva nfalgmrisa kevtnpvtge 481 piqlrvgiht gpvladvvgd kmpryclfgd tvntasrmes hglpnkvhls ptayralknq 541 gfkiiergei evkgkgrmtt yfliqnlnat edeimgrskt pvdhkgstqk aslpttklqg 601 svqpscpehs slaswll';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mygfvnhale llvirnygpe vwedikkeaq ldeegqflvr iiyddsktyd lvaaaskvln 61 lnageilqmf gkmffvfcqe sgydtilrvl gsnvreflqk neskhisdlh hcflgiylcc 121 nnldalhdhl atiypgmrap sfrctdaekg kglilhyyse reglqdivig iiktvaqqih 181 gteidmkviq qrneecdhtq flieekeske edfyedldrf eengtqesri spytfckafp 241 fhiifdrdlv vtqcgnaiyr vlpqlqpgnc sllsvfslvr phidisfhgi lshintvfvl 301 rskeglldve klecedeltg teisclrlkg qmiylpeads ilflcspsvm nlddltrrgl 361 ylsdiplhda trdlvllgeq freeykltqe leiltdrlql tlraledekk ktdtllysvl 421 ppsvanelrh krpvpakryd nvtilfsgiv gfnafcskha sgegamkivn llndlytrfd 481 tltdsrknpf vykvetvgdk ymtvsglpep cihharsich laldmmeiag qvqvdgesvq 541 itigihtgev vtgvigqrmp ryclfgntvn ltsrtettge kgkinvseyt yrclmspens 601 dpqfhlehrg pvsmkgkkep mqvwflsrkn tgteetkqdd d';

protein();

}

} else if (randomizer <= 60) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 15) {

peptidesequence = ' 1 mknniintqq sfvtmpnviv pdiekeirrm engacssfse dddsastsee senenpharg 61 sfsykslrkg gpsqreqylp gaialfnvnn ssnkdqepee kkkkkkekks ksddknenkn 121 dpekkkkkkd kekkkkeeks kdkkeeekke vvvidpsgnt yynwlfcitl pvmynwtmvi 181 aracfdelqs dyleywlild yvsdivylid mfvrtrtgyl eqgllvkeel klinkyksnl 241 qfkldvlsli ptdllyfklg wnypeirlnr llrfsrmfef fqrtetrtny pnifrisnlv 301 myiviiihwn acvfysiska igfgndtwvy pdindpefgr larkyvysly wstltlttig 361 etpppvrdse yvfvvvdfli gvlifativg nigsmisnmn aaraefqari daikqymhfr 421 nvskdmekrv ikwfdylwtn kktvdekevl kylpdklrae iainvhldtl kkvrifadce 481 agllvelvlk lqpqvyspgd yickkgdigr emyiikegkl avvaddgvtq fvvlsdgsyf 541 geisilnikg skagnrrtan iksigysdlf clskddlmea lteypdaktm leekgkqilm 601 kdglldlnia nagsdpkdle ekvtrmegsv dllqtrfari laeyesmqqk lkqrltkvek 661 flkplidtef ssiegpgaes gpidst';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mtektngvks spannhnhha ppaikangkd dhrtssrphs aadddtssel qrladvdapq 61 qgrsgfrriv rlvgiirewa nknfreeepr pdsflerfrg pelqtvttqe gdgkgdkdge 121 dkgtkkkfel fvldpagdwy ycwlfviamp vlynwcllva racfsdlqkg yylvwlvldy 181 vsdvvyiadl firlrtgfle qgllvkdtkk lrdnyihtlq fkldvasiip tdliyfavdi 241 hspevrfnrl lhfarmfeff drtetrtnyp nifrisnlvl yilviihwna ciyyaisksi 301 gfgvdtwvyp nitdpeygyl areyiyclyw stltlttige tpppvkdeey lfvifdflig 361 vlifativgn vgsmisnmna traefqakid avkhymqfrk vskgmeakvi rwfdylwtnk 421 ktvdereilk nlpaklraei ainvhlstlk kvrifhdcea gllvelvlkl rpqvfspgdy 481 icrkgdigke myiikegkla vvaddgvtqy allsagscfg eisilnikgs kmgnrrtani 541 rslgysdlfc lskddlmeav teypdakkvl eergreilmk eglldeneva tsmevdvqek 601 lgqletnmet lytrfgrlla eytgaqqklk qritvletkm kqnneddyls dgmnspelaa 661 adep';

protein();

} else if (randomizer <= 45) {

peptidesequence = '1 makintqysh psrthlkvkt sdrdlnraen glsrahssse etssvlqpgi ametrglads 61 gqgsftgqgi arlsrlifll rrwaarhvhh qdqgpdsfpd rfrgaelkev ssqesnaqan 121 vgsqepadrg rsawplakcn tntsnnteee kktkkkdaiv vdpssnlyyr wltaialpvf 181 ynwyllicra cfdelqseyl mlwlvldysa dvlyvldvlv rartgfleqg lmvsdtnrlw 241 qhyktttqfk ldvlslvptd laylkvgtny pevrfnrllk fsrlfeffdr tetrtnypnm 301 frignlvlyi liiihwnaci yfaiskfigf gtdswvypni sipehgrlsr kyiyslywst 361 ltlttigetp ppvkdeeylf vvvdflvgvl ifativgnvg smisnmnasr aefqakidsi 421 kqymqfrkvt kdletrvirw fdylwankkt vdekevlksl pdklkaeiai nvhldtlkkv 481 rifqdceagl lvelvlklrp tvfspgdyic kkgdigkemy iinegklavv addgvtqfvv 541 lsdgsyfgei silnikgsks gnrrtanirs igysdlfcls kddlmealte ypeakkalee 601 kgrqilmkdn lideelarag adpkdleekv eqlgssldtl qtrfarllae ynatqmkmkq 661 rlsqlesqvk gggdkpladg evpgdatkte dkqq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 msqdtkvktt essppapska rkllpvldps gdyyywwlnt mvfpvmynli ilvcracfpd 61 lqhgylvawl vldytsdlly lldmvvrfht gfleqgilvv dkgrissryv rtwsffldla 121 slmptdvvyv rlgphtptlr lnrflraprl feafdrtetr taypnafria klmlyifvvi 181 hwnsclyfal srylgfgrda wvypdpaqpg ferlrrqyly sfyfstlilt tvgdtpppar 241 eeeylfmvgd fllavmgfat imgsmssviy nmntadaafy pdhalvkkym klqhvnrkle 301 rrvidwyqhl qinkkmtnev ailqhlperl raevavsvhl stlsrvqifq nceaslleel 361 vlklqpqtys pgeyvcrkgd igqemyiire gqlavvaddg itqyavlgag lyfgeisiin 421 ikgnmsgnrr tanikslgys dlfclskedl revlseypqa qtimeekgre illkmnkldv 481 naeaaeialq eatesrlrgl dqqlddlqtk farllaeles salkiayrie rlewqtrewp 541 mpedlaeadd egepeegtsk deegrasqeg ppgpe';

protein();

} else if (randomizer <= 75) {

peptidesequence = '1 mlgwvqrvlp qppgtprktk mqeeeevepe pemeaevepe pnpeeaetes esmppeesfk 61 eeevavadps pqetkeaalt stislraqga eisemnspsr rvltwlmkgv ekvipqpvhs 121 itedpaqilg hgstgdtgct depnealeaq dtrpglrlll wleqnlervl pqppkssevw 181 rdepavatga asdpappgrp qemgpklqar etpslptpip lqpkeepkea papepqpgsq 241 aqtsslpptr dparlvawvl hrlemalpqp vlhgkigeqe pdspgicdvq tisilpggqv 301 epdlvleeve ppwedahqdv stspqgtevv payeeenkav ekmprelsri eeekedeeee 361 eeeeeeeeee evtevlldsc vvsqvgvgqs eedgtrpqst sdqklweevg eeakkeaeek 421 akeeaeevae eeaekepqdw aetkeepeae aeaassgvpa tkqhpevqve dtdadscplm 481 aeenppstvl pppspaksdt livpssasgt hrkklpsedd eaeelkalsp aespvvawsd 541 pttpkdtdgq draastastn saiindrlqe lvklfkerte kvkeklidpd vtsdeespkp 601 spakkapepa pdtkpaeaep veeehycdml cckfkhrpwk kyqfpqsidp ltnlmyvlwl 661 ffvvmawnwn cwlipvrwaf pyqtpdnihh wllmdylcdl iyflditvfq trlqfvrggd 721 iitdkkdmrn nylksrrfkm dllsllpldf lylkvgvnpl lrlprclkym affefnsrle 781 silskayvyr virttaylly slhlnsclyy wasayqglgs thwvydgvgn syircyyfav 841 ktlitigglp dpktlfeivf qllnyftgvf afsvmigqmr dvvgaatagq tyyrscmdst 901 vkymnfykip ksvqnrvktw yeytwhsqgm ldeselmvql pdkmrldlai dvnynivskv 961 alfqgcdrqm ifdmlkrlrs vvylpndyvc kkgeigremy iiqagqvqvl ggpdgksvlv 1021 tlkagsvfge isllavgggn rrtanvvahg ftnlfildkk dlneilvhyp esqkllrkka 1081 rrmlrsnnkp keeksvlilp pragtpklfn aalamtgkmg gkgakggkla hlrarlkela 1141 aleaaakqqe lveqakssqd vkgeegsaap dqhthpkeaa tdppaprtpp eppgsppssp 1201 ppaslgrpeg eeegpaepee hsvricmspg pepgeqilsv kmpeereeka e';

protein();

} else {

peptidesequence = '1 mfksltkvnk vkpigennen eqssrrneeg shpsnqsqqt taqeenkgee kslktkstpv 61 tseephtniq dklskknssg dlttnpdpqn aaeptgtvpe qkemdpgkeg pnspqnkppa 121 apvineyada qlhnlvkrmr qrtalykkkl vegdlsspea spqtakptav ppvkesddkp 181 tehyyrllwf kvkkmpltey lkriklpnsi dsytdrlyll wlllvtlayn wnccfiplrl 241 vfpyqtadni hywliadiic diiylydmlf iqprlqfvrg gdiivdsnel rkhyrtstkf 301 qldvasiipf dicylffgfn pmfranrmlk ytsffefnhh lesimdkayi yrvirttgyl 361 lfilhinacv yywasnyegi gttrwvydge gneylrcyyw avrtlitigg lpepqtlfei 421 vfqllnffsg vfvfssligq mrdvigaata nqnyfracmd dtiaymnnys ipklvqkrvr 481 twyeytwdsq rmldesdllk tlpttvqlal aidvnfsiis kvdlfkgcdt qmiydmllrl 541 ksvlylpgdf vckkgeigke myiikhgevq vlggpdgtkv lvtlkagsvf geisllaagg 601 gnrrtanvva hgfanlltld kktlqeilvh ypdserilmk karvllkqka ktaeatpprk 661 dlallfppke etpklfktll ggtgkaslar llklkreqaa qkkensegge eegkenedkq 721 kenedkqken edkgkenedk dkgrepeekp ldrpectasp iaveeephsv rrtvlprgts 781 rqsliismap saeggeevlt ievkekakq';

protein();

}

} else {

heme();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.05 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mahrfpaltq eqkkelseia qsivangkgi laadesvgtm gnrlqrikve nteenrrqfr 61 eilfsvdssi nqsiggvilf hetlyqkdsq gklfrnilke kgivvgikld qggaplagtn 121 kettiqgldg lsercaqykk dgvdfgkwra vlriadqcps slaiqenana laryasicqq 181 nglvpivepe vipdgdhdle hcqyvtekvl aavykalndh hvylegtllk pnmvtaghac 241 tkkytpeqva matvtalhrt vpaavpgicf lsggmseeda tlnlnainlc plpkpwklsf 301 sygralqasa laawggkaan keatqeafmk ramancqaak gqyvhtgssg aastqslfta 361 cyty';

protein();

} else {

peptidesequence = '1 mphsypalsa eqkkelsdia lrivapgkgi laadesvgsm akrlsqigve nteenrrlyr 61 qvlfsaddrv kkciggviff hetlyqkddn gvpfvrtiqd kgivvgikvd kgvvplagtd 121 getttqgldg lsercaqykk dgadfakwrc vlkisertps alailenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hvylegtllk pnmvtpghac 241 pikytpeeia matvtalrrt vppavpgvtf lsggqseeea sfnlnainrc plprpwaltf 301 sygralqasa lnawrgqrdn agaateefik raevnglaaq gkyegsgedg gaaaqslyia 361 nhay';

protein();

}

} else if (randomizer <= 50) {

peptidesequence = ' 1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mslsnkltld kldvkgkrvv mrvdfnvpmk nnqitnnqri kaavpsikfc ldngaksvvl 61 mshlgrpdgv pmpdkyslep vavelksllg kdvlflkdcv gpevekacan paagsville 121 nlrfhveeeg kgkdasgnkv kaepakieaf raslsklgdv yvndafgtah rahssmvgvn 181 lpqkaggflm kkelnyfaka lesperpfla ilggakvadk iqlinnmldk vnemiigggm 241 aftflkvlnn meigtslfde egakivkdlm skaekngvki tlpvdfvtad kfdenaktgq 301 atvasgipag wmgldcgpes skkyaeavtr akqivwngpv gvfeweafar gtkalmdevv 361 katsrgciti igggdtatcc akwntedkvs hvstgggasl ellegkvlpg vdalsni';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mvkivtvktq ayqdqkpgts glrkrvkvfq ssanyaenfi qsiistvepa qrqeatlvvg 61 gdgrfymkea iqliariaaa ngigrlvigq ngilstpavs ciirkikaig giiltashnp 121 ggpngdfgik fnisnggpap eaitdkifqi sktieeyavc pdlkvdlgvl gkqqfdlenk 181 fkpftveivd sveayatmlr sifdfsalke llsgpnrlki ridamhgvvg pyvkkilcee 241 lgapansavn cvpledfggh hpdpnltyaa dlvetmksge hdfgaafdgd gdrnmilgkh 301 gffvnpsdsv aviaanifsi pyfqqtgvrg farsmptsga ldrvasatki alyetptgwk 361 ffgnlmdask lslcgeesfg tgsdhirekd glwavlawls ilatrkqsve dilkdhwqky 421 grnfftrydy eeveaegank mmkdlealmf drsfvgkqfs andkvytvek adnfeysdpv 481 dgsisrnqgl rliftdgsri vfrlsgtgsa gatirlyids yekdvakinq dpqvmlapli 541 sialkvsqlq ertgrtaptv it';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 msiekiware ildsrgnptv evdlytakgl fraavpsgas tgiyealelr dgdkqrylgk 61 gvlkavdhin stiapaliss glsvveqekl dnlmleldgt enkskfgana ilgvslavck 121 agaaerelpl yrhiaqlagn sdlilpvpaf nvinggshag nklamqefmi lpvgaesfrd 181 amrlgaevyh tlkgvikdky gkdatnvgde ggfapnilen sealelvkea idkagyteki 241 vigmdvaase fyrdgkydld fksptdpsry itgdqlgaly qdfvrdypvv siedpfdqdd 301 waawskftan vgiqivgddl tvtnpkrier aveekacncl llkvnqigsv teaiqackla 361 qengwgvmvs hrsgetedtf iadlvvglct gqiktgapcr serlakynql mrieeelgde 421 arfaghnfrn psvl';

protein();

} else {

peptidesequence = ' 1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.1 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

} else {

peptidesequence = ' 1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.15 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mefpigslet nnfrrftpes lveiekqiaa kqgtkkarek hreqkdqeek prpqldlkac 61 nqlpkfygel paeligeple dldpfysthr tfmvlnkgrt isrfsatral wlfspfnlir 121 rtaikvsvhs wfslfitvti lvncvcmtrt dlpekieyvf tviytfeali kilargfcln 181 eftylrdpwn wldfsvitla yvgtaidlrg isglrtfrvl ralktvsvip glkvivgali 241 hsvkkladvt iltifclsvf alvglqlfkg nlknkcvknd mavnettnys shrkpdiyin 301 krgtsdpllc gngsdsghcp dgyiclktsd npdfnytsfd sfawaflslf rlmtqdswer 361 lyqqtlrtsg kiymiffvlv iflgsfylvn lilavvtmay eeqnqattde ieakekkfqe 421 alemlrkeqe vlaalgidtt slhshngspl tsknaserrh rikprvsegs tednksprsd 481 pynqrrmsfl glasgkrras hgsvfhfrsp grdislpegv tddgvfpgdh eshrgslllg 541 ggagqqgplp rsplpqpsnp dsrhgedehq ppptselapg avdvsafdag qkktflsaey 601 ldepfraqra msvvsiitsv leeleeseqk cppcltslsq kyliwdccpm wvklktilfg 661 lvtdpfaelt itlcivvnti fmamehhgms ptfeamlqig nivftiffta emvfkiiafd 721 pyyyfqkkwn ifdciivtvs llelgvakkg slsvlrsfrl lrvfklaksw ptlntlikii 781 gnsvgalgnl tiilaiivfv falvgkqllg enyrnnrkni saphedwprw hmhdffhsfl 841 ivfrilcgew ienmwacmev gqksiclilf ltvmvlgnlv vlnlfialll nsfsadnlta 901 peddgevnnl qvalariqvf ghrtkqalcs ffsrscpfpq pkaepelvvk lplssskaen 961 hiaantargs sgglqaprgp rdehsdfian ptvwvsvpia egesdlddle ddggedaqsf 1021 qqevipkgqq eqlqqvercg dhltprspgt gtssedlaps lgetwkdesv pqvpaegvdd 1081 tsssegstvd cldpeeilrk ipeladdlee pddcftegci rhcpcckldt tkspwdvgwq 1141 vrktcyrive hswfesfiif millssgsla fedyyldqkp tvkalleytd rvftfifvfe 1201 mllkwvaygf kkyftnawcw ldflivnisl isltakiley sevapikalr tlralrplra 1261 lsrfegmrvv vdalvgaips imnvllvcli fwlifsimgv nlfagkfwrc inytdgefsl 1321 vplsivnnks dckiqnstgs ffwvnvkvnf dnvamgylal lqvatfkgwm dimyaavdsr 1381 evnmqpkwed nvymylyfvi fiifggfftl nlfvgviidn fnqqkkklgg qdifmteeqk 1441 kyynamkklg skkpqkpipr plnkfqgfvf divtrqafdi timvliclnm itmmvetddq 1501 seektkilgk inqffvavft gecvmkmfal rqyyftngwn vfdfivvvls iaslifsail 1561 kslqsyfspt lfrvirlari grilrliraa kgirtllfal mmslpalfni glllflvmfi 1621 ysifgmssfp hvrweagidd mfnfqtfans mlclfqitts agwdgllspi lntgppycdp 1681 nlpnsngtrg dcgspavgii ffttyiiisf limvnmyiav ilenfnvate esteplsedd 1741 fdmfyetwek fdpeatqfit fsalsdfadt lsgplripkp nrniliqmdl plvpgdkihc 1801 ldilfaftkn vlgesgelds lkanmeekfm atnlskssye piattlrwkq edisatviqk 1861 ayrsyvlhrs malsntpcvp raeeeaaslp degfvaftan encvlpdkse tasatsfpps 1921 yesvtrglsd rvnmrtsssi qnedeatsme liapgp';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+28);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1.2 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = 10;

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 6.12604808e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvatgdpad eaaalpghpq dtydpeadhe ccervvinis glrfetqlkt laqfpetllg 61 dpkkrmryfd plrneyffdr nrpsfdaily yyqsggrlrr pvnvpldifs eeirfyelge 121 eamemfrede gyikeeerpl penefqrqvw llfeypessg pariiaivsv mvilisivsf 181 cletlpifrd enedmhgsgv tfhtysnsti gyqqstsftd pffivetlci iwfsfeflvr 241 ffacpskagf ftnimniidi vaiipyfitl gtelaekped aqqgqqamsl ailrvirlvr 301 vfrifklsrh skglqilgqt lkasmrelgl lifflfigvi lfssavyfae aderesqfps 361 ipdafwwavv smttvgygdm vpttiggkiv gslcaiagvl tialpvpviv snfnyfyhre 421 tegeeqaqyl qvtscpkips spdlkksrsa stisksdyme iqegvnnsne dfreenlkta 481 nctlantnyv nitkmltdv';

protein();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 3.093963676e+30);

cursor1y = spawnery - 1 \* 3.093963676e+28;

cursor1x = spawnerx - 1 \* 3.093963676e+28;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round((18 \* 3.093963676e+28) / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round((18 \* 3.093963676e+28) / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round((1 \* 3.06302404e+30) / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 90) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

peptidesequence = '1 mngtegpnfy vpfsnatgvv rspfeypqyy laepwqfsml aaymfllivl gfpinfltly 61 vtvqhkklrt plnyillnla vadlfmvlgg ftstlytslh gyfvfgptgc nlegffatlg 121 geialwslvv laieryvvvc kpmsnfrfge nhaimgvaft wvmalacaap plagwsryip 181 eglqcscgid yytlkpevnn esfviymfvv hftipmiiif fcygqlvftv keaaaqqqes 241 attqkaekev trmviimvia flicwvpyas vafyifthqg snfgpifmti paffaksaai 301 ynpviyimmn kqfrncmltt iccgknplgd deasatvskt etsqvapa';

protein();

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

peptidesequence = '1 mgagasaeek hsrelekklk edaekdartv kllllgages gkstivkqmk iihqdgysle 61 eclefiaiiy gntlqsilai vramttlniq ygdsarqdda rklmhmadti eegtmpkems 121 diiqrlwkds giqacferas eyqlndsagy ylsdlerlvt pgyvpteqdv lrsrvkttgi 181 ietqfsfkdl nfrmfdvggq rserkkwihc fegvtciifi aalsaydmvl veddevnrmh 241 eslhlfnsic nhryfattsi vlflnkkdvf fekikkahls icfpdydgpn tyedagnyik 301 vqflelnmrr dvkeiyshmt catdtqnvkf vfdavtdiii kenlkdcglf';

} else if (randomizer <= 60) {

peptidesequence = '1 mgsgasaedk elakrskele kklqedadke aktvkllllg agesgkstiv kqmkiihqdg 61 yspeeclefk aiiygnvlqs ilaiiramtt lgidyaepsc addgrqlnnl adsieegtmp 121 pelvevirrl wkdggvqacf eraaeyqlnd sasyylnqle ritdpeylps eqdvlrsrvk 181 ttgiietkfs vkdlnfrmfd vggqrserkk wihcfegvtc iifcaalsay dmvlveddev 241 nrmheslhlf nsicnhkffa atsivlflnk kdlfeekikk vhlsicfpey dgnnsyddag 301 nyiksqfldl nmrkdvkeiy shmtcatdtq nvkfvfdavt diiikenlkd cglf';

} else {

peptidesequence = '1 mgsgissesk esakrskele kklqedaerd artvkllllg agesgkstiv kqmkiihkng 61 yseqecmefk aviysntlqs ilaivkamtt lgidyvnprs aedqrqlyam antledggmt 121 pqlaevikrl wrdpgiqacf eraseyqlnd saayylndld ritasgyvpn eqdvlhsrvk 181 ttgiietqfs fkdlhfrmfd vggqrserkk wihcfegvtc iifcaalsay dmvlvedeev 241 nrmheslhlf nsicnhkyfs ttsivlflnk kdifqekvtk vhlsicfpey tgpntfedag 301 nyiknqfldl nlkkedkeiy shmtcatdtq nvkfvfdavt diiikenlkd cglf';

}

protein();

} else if (randomizer <= 75) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 15) {

peptidesequence = '1 mgevtaeeve kfldsnigfa kqyynlhyra klisdllgak eaavdfsnyh spssmeesei 61 ifdllrdfqe nlqtekcifn vmkklcfllq adrmslfmyr trngiaelat rlfnvhkdav 121 ledclvmpdq eivfpldmgi vghvahskki anvpnteede hfcdfvdilt eyktknilas 181 pimngkdvva iimavnkvdg shftkrdeei llkylnfanl imkvyhlsyl hncetrrgqi 241 llwsgskvfe eltdierqfh kalytvrafl ncdrysvgll dmtkqkeffd vwpvlmgevp 301 pysgprtpdg reinfykvid yilhgkedik vipnpppdhw alvsglpayv aqnglicnim 361 napaedffaf qkepldesgw miknvlsmpi vnkkeeivgv atfynrkdgk pfdemdetlm 421 esltqflgws vlnpdtyesm nklenrkdif qdivkyhvkc dneeiqkilk trevygkepw 481 eceeeelaei lqaelpdadk yeinkfhfsd lpltelelvk cgiqmyyelk vvdkfhipqe 541 alvrfmysls kgyrkityhn wrhgfnvgqt mfsllvtgkl kryftdleal amvtaafchd 601 idhrgtnnly qmksqnplak lhgssilerh hlefgktllr deslnifqnl nrrqhehaih 661 mmdiaiiatd lalyfkkrtm fqkivdqskt yeseqewtqy mmleqtrkei vmammmtacd 721 lsaitkpwev qsqvallvaa efweqgdler tvlqqnpipm mdrnkadelp klqvgfidfv 781 ctfvykefsr fheeitpmld gitnnrkewk aladeydakm kvqeekkqkq qsaksaaagn 841 qpggnpspgg attskscciq';

} else if (randomizer <= 30) {

peptidesequence = '1 mslseeqars fldqnpdfar qyfgkklspe nvaaacedgc ppdcdslrdl cqveestall 61 elvqdmqesi nmervvfkvl rrlctllqad rcslfmyrqr ngvaelatrl fsvqpdsvle 121 dclvppdsei vfpldigvvg hvaqtkkmvn vedvaecphf ssfadeltdy ktknmlatpi 181 mngkdvvavi mavnklngpf ftsededvfl kylnfatlyl kiyhlsylhn cetrrgqvll 241 wsankvfeel tdierqfhka fytvraylnc erysvglldm tkekeffdvw svlmgesqpy 301 sgprtpdgre ivfykvidyv lhgkeeikvi ptpsadhwal asglpsyvae sgficnimna 361 sademfkfqe galddsgwli knvlsmpivn kkeeivgvat fynrkdgkpf deqdevlmes 421 ltqflgwsvm ntdtydkmnk lenrkdiaqd mvlyhvkcdr deiqlilptr arlgkepadc 481 dedelgeilk eelpgpttfd iyefhfsdle cteldlvkcg iqmyyelgvv rkfqipqevl 541 vrflfsiskg yrrityhnwr hgfnvaqtmf tllmtgklks yytdleafam vtaglchdid 601 hrgtnnlyqm ksqnplaklh gssilerhhl efgkfllsee tlniyqnlnr rqhehvihlm 661 diaiiatdla lyfkkramfq kivdesknyq dkkswveyls lettrkeivm ammmtacdls 721 aitkpwevqs kvallvaaef weqgdlertv ldqqpipmmd rnkaaelpkl qvgfidfvct 781 fvykefsrfh eeilpmfdrl qnnrkewkal adeyeakvka leekeeeerv aakkvgteic 841 nggpapksst ccil';

} else if (randomizer <= 45) {

peptidesequence = '1 mgeinqvave kyleenpqfa keyfdrklrv evlgeifkns qvpvqssmsf seltqveesa 61 lclellwtvq eeggtpeqgv hralqrlahl lqadrcsmfl crsrngipev asrlldvtpt 121 skfednlvgp dkevvfpldi givgwaahtk kthnvpdvkk nshfsdfmdk qtgyvtknll 181 atpivvgkev lavimavnkv nasefskqde evfskylnfv siilrlhhts ymyniesrrs 241 qilmwsankv feeltdverq fhkalytvrs ylncerysig lldmtkekef ydewpiklge 301 vepykgpktp dgrevnfyki idyilhgkee ikviptppad hwtlisglpt yvaengficn 361 mmnapadeyf tfqkgpvdet gwviknvlsl pivnkkediv gvatfynrkd gkpfdehdey 421 itetltqflg wsllntdtyd kmnklenrkd iaqemlmnqt katpeeiksi lkfqeklnvd 481 viddceekql vailkedlpd prsaelyefr fsdfpltehg likcgirlff einvvekfkv 541 pvevltrwmy tvrkgyravt yhnwrhgfnv gqtmftllmt grlkkyytdl eafamlaaaf 601 chdidhrgtn nlyqmkstsp larlhgssil erhhleyskt llqdeslnif qnlnkrqfet 661 vihlfevaii atdlalyfkk rtmfqkivda ceqmqteeea ikyvtvdptk keiimammmt 721 acdlsaitkp wevqsqvalm vanefweqgd lertvlqqqp ipmmdrnkrd elpklqvgfi 781 dfvctfvyke fsrfhkeitp mlsglqnnrv ewksladeyd akmkvieeea kkqeggaeka 841 aedsgggddk ksktclml';

} else if (randomizer <= 60) {

peptidesequence = '1 msdnttlpap asnqgpttpr kgppkfkqrq trqfkskppk kgvkgfgddi pgmeglgtdi 61 tvicpweafs hlelhelaqf gii';

} else if (randomizer <= 75) {

peptidesequence = '1 mnleppkaef rsatrvaggp vtprkgppkf kqrqtrqfks kppkkgvqgf gddipgmegl 61 gtditvicpw eafnhlelhe laqygii';

} else {

peptidesequence = '1 mnleppkaef rsatrvaggp vtprkgppkf kqrqtrqfks kppkkgvqgf gddipgmegl 61 gtditvicpw eafnhlelhe laqygii';

}

protein();

} else {

retinal();

}

} else {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 20) {

peptidesequence = ' 1 msiqvehpag gykklfetve elsspltahv tgriplwltg sllrcgpglf evgsepfyhl 61 fdgqallhkf dfkeghvtyh rrfirtdayv ramtekrivi tefgtcafpd pcknifsrff 121 syfrgvevtd nalvnvypvg edyyactetn fitkinpetl etikqvdlcn yvsvngatah 181 phiendgtvy nigncfgknf siaynivkip plqadkedpi skseivvqfp csdrfkpsyv 241 hsfgltpnyi vfvetpvkin lfkflsswsl wganymdcfe snetmgvwlh iadkkrkkyl 301 nnkyrtspfn lfhhintyed ngflivdlcc wkgfefvyny lylanlrenw eevkknarka 361 pqpevrryvl plnidkadtg knlvtlpntt atailcsdet iwlepevlfs gprqafefpq 421 inyqkycgkp ytyayglgln hfvpdrlckl nvktketwvw qepdsypsep ifvshpdale 481 eddgvvlsvv vspgagqkpa yllilnakdl sevaraevei nipvtfhglf kks';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mknpmlevvs llleklllis nftlfssgaa gedkgrnsfy etssfhrgdv levprthlth 61 ygiylgdnrv ahmmpdilla ltddmgrtqk vvsnkrlilg vivkvasirv dtvedfayga 121 nilvnhldes lqkkallnee varraekllg ftpysllwnn cehfvtycry gtpispqsdk 181 fcetvkiiir dqrsvlasav lglasivctg lvsyttlpai fipfflwmag';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mwlplllgal lwavlwllrd rqslpasnaf vfitgcdsgf grllalqldq rgfrvlascl 61 tpsgaedlqr vassrlhttl lditdpqsvq qaakwvemhv keaglfglvn nagvagiigp 121 tpwltrddfq rvlnvntmgp igvtlallpl lqqargrvin itsvlgrlaa ngggycvskf 181 gleafsdslr rdvahfgirv sivepgffrt pvtnleslek tlqacwarlp patqahygga 241 fltkylkmqq rimnlicdpd ltkvsrcleh altarhprtr yspgwdakll wlpasylpas 301 lvdavltwvl pkpaqavy';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mlvtlgllts ffsflymvap sirkffaggv crtnvqlpgk vvvitgantg igketarela 61 srgarvyiac rdvlkgesaa seirvdtkns qvlvrkldls dtksirafae gflaeekqlh 121 ilinnagvmm cpysktadgf ethlgvnhlg hflltyllle rlkvsaparv vnvssvahhi 181 gkipfhdlqs ekrysrgfay chsklanvlf trelakrlqg tgvttyavhp gvvrselvrh 241 ssllcllwrl fspfvktare gaqtslhcal aegleplsgk yfsdckrtwv sprarnnkta 301 erlwnvscel lgirwe';

protein();

} else {

iron\_cofactor();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((20 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 3.06302404e+30) / density\_parameter\_z - 8);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((20 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 3.06302404e+30) / density\_parameter\_z - 8);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 5.569134618e+29;

cursor1x = spawnerx - 1 \* 1;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((20 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((1 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 3.06302404e+30) / density\_parameter\_z - 8);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 1 \* (1 \* 1);

cursor1y = spawnery - 1 \* 1;

cursor1x = spawnerx - 1 \* 5.569134618e+29;

density\_parameter\_x = 1.23758547e+26 \* 1;

density\_parameter\_y = 1.23758547e+26 \* 1;

density\_parameter\_z = 1.23758547e+26;

generation\_parameter\_x = Math.round((1 \* 3.093963676e+28) / density\_parameter\_x - 8);

generation\_parameter\_3 = Math.round((20 \* 3.093963676e+28) / density\_parameter\_y - 8);

generation\_parameter\_z = Math.round((1 \* 3.06302404e+30) / density\_parameter\_z - 8);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

collagen();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((5 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((10 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.9 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((4.8 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow(((9.95 \* 6.0221409e+23) / 1000) / 1000000000000, 1 / 3);

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1z = loc5[0];

cursor1y = loc5[1] - 0;

cursor1x = loc5.slice(-1)[0] - 0;

if (unused\_variable == 5) {

cursor1x = spawnerx - 1;

cursor1y = spawnery - 1;

cursor1z = spawnerz + 1;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

if (unused\_variable == 5) {

generation\_parameter\_x = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.18792735e+29 / density\_parameter\_x - 0);

generation\_parameter\_z = Math.round(6.187927352e+30 / density\_parameter\_x - 0);

}

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_cytosol();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cells\_list.push(['rod cell', spawnerz, spawnery, spawnerx, spawnerz + 6.187927352e+30, spawnery - 6.187927352e+29, spawnerx - 6.187927352e+29]);

}

}

function default\_cell\_fluids() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((5 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((10 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 0 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.9 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 0 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 0 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((9.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 0 \* 1;

cursor1y = spawnery - 0 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.8 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 0 \* 1;

cursor1z = spawnerz + 0 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 0 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 0 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (cursor1x < spawnerx - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1y < spawnery - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1z > spawnerz + 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1x > spawnerx - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1y > spawnery - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1z < spawnerz + 3.093963676e+28 \* 1) {

randomize\_ECF();

} else {

randomize\_cytosol();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function supported\_cell() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.floor(5.878530986e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.floor(5.878530986e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_3; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_3; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + (6.18792735e+29 - 4.085906753e+25)) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 3.15584295e+28 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

nucleoid\_size\_z = 5.507255344e+29;

nucleoid\_size\_y = 4.331549147e+29;

nucleoid();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.25973825e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 8; countcell1++) {

mitochondrion();

cursor1x = cursor1x - 6.187927353e+28;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

vital\_components();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 1.299464744e+29 \* 1;

vital\_components2();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 2.475170941e+29 \* 1;

vital\_components3();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((5 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((10 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.9 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((9.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.8 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (cursor1y < spawnery - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1z < spawnerz + 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1x > spawnerx - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1y > spawnery - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1x < spawnerx - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else {

randomize\_cytosol();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function supporting\_cell() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z <= spawnerz + 5.878530986e+29) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 1 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z <= spawnerz + 5.878530986e+29) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 1 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z <= spawnerz + 5.878530986e+29) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 1 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

while (cursor1z <= spawnerz + 5.878530986e+29) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 1 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 5.878530986e+29 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_3; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (countcell1 = 0; countcell1 < generation\_parameter\_x; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_3; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + 5.878530986e+29) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + 5.878530986e+29) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + 5.878530986e+29) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

while (cursor1z < spawnerz + 5.878530986e+29) {

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 3.15584295e+28 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

nucleoid\_size\_z = 5.507255344e+29;

nucleoid\_size\_y = 4.331549147e+29;

genomeprevious = genome;

genome = pelagibacter\_genome;

nucleoid();

genome = genomeprevious;

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.25973825e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 8; countcell1++) {

mitochondrion();

cursor1x = cursor1x - 6.187927353e+28;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

vital\_components();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 1.299464744e+29 \* 1;

vital\_components2();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 2.475170941e+29 \* 1;

vital\_components3();

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((5 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((10 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.9 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((9.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.187927353e+29 / Math.pow((4.8 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(6.187927353e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(6.187927353e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(6.187927353e+29 / density\_parameter\_z - 3);

for (countcell1 = 0; countcell1 < generation\_parameter\_3; countcell1++) {

for (countcell2 = 0; countcell2 < generation\_parameter\_x; countcell2++) {

for (countcell3 = 0; countcell3 < generation\_parameter\_z; countcell3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (cursor1x < spawnerx - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1y < spawnery - 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1z > spawnerz + 5.878530986e+29 \* 1) {

randomize\_ECF();

} else if (cursor1x > spawnerx - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1y > spawnery - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else {

randomize\_cytosol();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

}

function acinicyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count2 = 0; count2 < 1; count2++) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mngeeqyyaa tqlykdpcaf qrgpapefsa sppaclymgr qpppppphpf pgalgaleqg 61 sppdispyev ppladdpava hlhhhlpaql alphppagpf pegaepgvle epnrvqlpfp 121 wmkstkahaw kgqwaggaya aepeenkrtr taytraqlle lekeflfnky isrprrvela 181 vmlnlterhi kiwfqnrrmk wkkeedkkrg ggtavggggv aepeqdcavt sgeellalpp 241 ppppggavpp aapvaaregr lppglsaspq pssvaprrpq epr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count4 = 0; count4 < 1; count4++) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mngeeqyyaa tqlykdpcaf qrgpapefsa sppaclymgr qpppppphpf pgalgaleqg 61 sppdispyev ppladdpava hlhhhlpaql alphppagpf pegaepgvle epnrvqlpfp 121 wmkstkahaw kgqwaggaya aepeenkrtr taytraqlle lekeflfnky isrprrvela 181 vmlnlterhi kiwfqnrrmk wkkeedkkrg ggtavggggv aepeqdcavt sgeellalpp 241 ppppggavpp aapvaaregr lppglsaspq pssvaprrpq epr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count6 = 0; count6 < 1; count6++) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlavgamegt rqsafllssp plaalhsmae mktplypaay pplpagppss sssssssssp 61 spplgthnpg glkppatggl sslgsppqql saatphgind ilsrpsmpva sgaalpsasp 121 sgssssssss asassasaaa aaaaaaaaaa sspagllagl prfsslsppp pppglyfsps 181 aaavaavgry pkplaelpgr tpifwpgvmq sppwrdarla ctphqgsill dkdgkrkhtr 241 ptfsgqqifa lektfeqtky lagperarla yslgmtesqv kvwfqnrrtk wrkkhaaema 301 takkkqdset erlkgasene eedddynkpl dpnsddekit qllkkhksss ggggglllha 361 sepesss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count8 = 0; count8 < 1; count8++) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlavgamegt rqsafllssp plaalhsmae mktplypaay pplpagppss sssssssssp 61 spplgthnpg glkppatggl sslgsppqql saatphgind ilsrpsmpva sgaalpsasp 121 sgssssssss asassasaaa aaaaaaaaaa sspagllagl prfsslsppp pppglyfsps 181 aaavaavgry pkplaelpgr tpifwpgvmq sppwrdarla ctphqgsill dkdgkrkhtr 241 ptfsgqqifa lektfeqtky lagperarla yslgmtesqv kvwfqnrrtk wrkkhaaema 301 takkkqdset erlkgasene eedddynkpl dpnsddekit qllkkhksss ggggglllha 361 sepesss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['acinicyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function adipocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count10 = 0; count10 < 1; count10++) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mipgnrmlmv vllcqvllgg ashaslipet gkkkvaeiqg haggrrsgqs hellrdfeat 61 llqmfglrrr pqpsksavip dymrdlyrlq sgeeeeeqih stgleyperp asrantvrsf 121 hheehlenip gtsensafrf lfnlssipen evissaelrl freqvdqgpd wergfhrini 181 yevmkppaev vpghlitrll dtrlvhhnvt rwetfdvspa vlrwtrekqp nyglaievth 241 lhqtrthqgq hvrisrslpq gsgnwaqlrp llvtfghdgr ghaltrrrra krspkhhsqr 301 arkknkncrr hslyvdfsdv gwndwivapp gyqafychgd cpfpladhln stnhaivqtl 361 vnsvnssipk accvptelsa ismlyldeyd kvvlknyqem vvegcgcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count12 = 0; count12 < 1; count12++) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maspskgndl fspdeegpav vagpgpgpgg aegaaeerrv kvsslpfsve almsdkkppk 61 easplpaesa sagatlrpll lsghgareah spgplvkpfe tasvksense dgaawmqepg 121 rysppprhms pttctlrkhk tnrkprtpft tsqllalerk frqkqylsia eraefsssln 181 ltetqvkiwf qnrrakakrl qeaeleklkm aakpmlpssf slpfpisspl qaasiygasy 241 pfhrpvlpip pvglyatpvg ygmyhls';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count14 = 0; count14 < 1; count14++) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msatgpisny yvdslishdn edllasrfpa tgahpaaarp sglvpdcsdf pscsfapkpa 61 vfstswapvp sqssvvyhpy gpqphlgadt rymrtwlepl sgavsfpsfp aggrhyalkp 121 daypgrradc gpgegrsypd ymygspgelr drapqtlpsp eadalagskh keekadldps 181 npvanwihar strkkrcpyt kyqtleleke flfnmyltrd rryevarvln lterqvkiwf 241 qnrrmkmkkm nkektdkeqs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count16 = 0; count16 < 1; count16++) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mssyfvnsfc grypngpdyq lhnygdhssv seqfrdsasm hsgrygygyn gmdlsvgrsg 61 sghfgsgera rsyaasasaa paeprysqpa tsthspqpdp lpcsavapsp gsdshhggkn 121 slsnssgasa dagsthissr egvgtasgae edapasseqa saqsepspap paqpqiypwm 181 rklhishdni ggpegkrart aytryqtlel ekefhfnryl trrrrieiah alclserqik 241 iwfqnrrmkw kkdnklksms maaaggafrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['adipocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function adrenocyte\_1() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count18 = 0; count18 < 1; count18++) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdysydedld elcpvcgdkv sgyhyglltc esckgffkrt vqnnkhytct esqsckidkt 61 qrkrcpfcrf qkcltvgmrl eavradrmrg grnkfgpmyk rdralkqqkk aqirangfkl 121 etgppmgvpp ppppapdyvl ppslhgpepk glaagppagp lgdfgapalp mavpgahgpl 181 agylypafpg raikseypep yasppqpglp ygypepfsgg pnvpelilql lqlepdedqv 241 rarilgclqe ptksrpdqpa afgllcrmad qtfisivdwa rrcmvfkele vadqmtllqn 301 cwsellvfdh iyrqvqhgke gsillvtgqe velttvatqa gsllhslvlr aqelvlqlla 361 lqldrqefvc lkfiilfsld lkflnnhilv kdaqekanaa lldytlchyp hcgdkfqqll 421 lclvevrals mqakeylyhk hlgnemprnn lliemlqakq t';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count20 = 0; count20 < 1; count20++) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mamvvstwrd pqdevpgsqg sqasqappvp gpppgaphtp qtpgqggpas tpaqtaaggq 61 ggpggpgsdk qqqqqhiecv vcgdkssgkh ygqftcegck sffkrsvrrn lsytcranrn 121 cpidqhhrnq cqycrlkkcl kvgmrreavq rgrmpptqpt hgqfaltngd plnchsylsg 181 yislllraep yptsrfgsqc mqpnnimgie nicelaarml fsavewarni pffpdlqitd 241 qvallrltws elfvlnaaqc smplhvapll aaaglhaspm sadrvvafmd hirifqeqve 301 klkalhvdsa eysclkaivl ftsdacglsd vahveslqek sqcaleeyvr sqypnqptrf 361 gklllrlpsl rtvsssvieq lffvrlvgkt pietlirdml lsgssfnwpy maiq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count22 = 0; count22 < 1; count22++) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavqesaaql smtlkvqeyp tlkvpyetln krfraaqkni dretshvtmv vaelektlsg 61 cpavdsvvsl ldgvveklsv lkrkavesiq aedesaklck rriehlkehs sdqpaaasvw 121 krkrmdrmmv ehllrcgyyn tavklarqsg iedlvniemf ltakeveesl erretatcla 181 wchdnksrlr kmksclefsl riqefielir qnkrldavrh arkhfsqaeg sqldevrqam 241 gmlafppdth ispykdlldp arwrmliqqf rydnyrlhql gnnsvftltl qaglsaiktp 301 qcykedgssk spdcpvcsrs lnklaqplpm ahcansrlvc kisgdvmnen nppmmlpngy 361 vygynsllsi rqddkvvcpr tkevfhfsqa ekvyim';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count24 = 0; count24 < 1; count24++) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgtshpaflv lgclltglsl ilcqlslpsi lpnenekvvq lnssfslrcf gesevswqyp 61 mseeessdve irneennsgl fvtvlevssa saahtglytc yynhtqteen elegrhiyiy 121 vpdpdvafvp lgmtdylviv edddsaiipc rttdpetpvt lhnsegvvpa sydsrqgfng 181 tftvgpyice atvkgkkfqt ipfnvyalka tseldlemea lktvyksget ivvtcavfnn 241 evvdlqwtyp gevkgkgitm leeikvpsik lvytltvpea tvkdsgdyec aarqatrevk 301 emkkvtisvh ekgfieikpt fsqleavnlh evkhfvvevr aypppriswl knnltlienl 361 teittdveki qeiryrsklk lirakeedsg hytivaqned avksytfell tqvpssildl 421 vddhhgstgg qtvrctaegt plpdiewmic kdikkcnnet swtilannvs niiteihsrd 481 rstvegrvtf akveetiavr claknllgae nrelklvapt lrseltvaaa vlvllvivii 541 slivlvviwk qkpryeirwr viesispdgh eyiyvdpmql pydsrwefpr dglvlgrvlg 601 sgafgkvveg tayglsrsqp vmkvavkmlk ptarssekqa lmselkimth lgphlnivnl 661 lgactksgpi yiiteycfyg dlvnylhknr dsflshhpek pkkeldifgl npadestrsy 721 vilsfenngd ymdmkqadtt qyvpmlerke vskysdiqrs lydrpasykk ksmldsevkn 781 llsddnsegl tlldllsfty qvargmefla skncvhrdla arnvllaqgk ivkicdfgla 841 rdimhdsnyv skgstflpvk wmapesifdn lyttlsdvws ygillweifs lggtpypgmm 901 vdstfynkik sgyrmakpdh atsevyeimv kcwnsepekr psfyhlseiv enllpgqykk 961 syekihldfl ksdhpavarm rvdsdnayig vtykneedkl kdweggldeq rlsadsgyii 1021 plpdidpvpe eedlgkrnrh ssqtseesai etgsssstfi kredetiedi dmmddigids 1081 sdlvedsfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['adrenocyte 1', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function adrenocyte\_2() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count26 = 0; count26 < 1; count26++) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 messakmesg gagqqpqpqp qqpflppaac ffataaaaaa aaaaaaaqsa qqqqqqqqqq 61 qqapqlrpaa dgqpsggghk sapkqvkrqr ssspelmrck rrlnfsgfgy slpqqqpaav 121 arrnerernr vklvnlgfat lrehvpngaa nkkmskvetl rsaveyiral qqlldehdav 181 saafqagvls ptispnysnd lnsmagspvs syssdegsyd plspeeqell dftnwf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count28 = 0; count28 < 1; count28++) {

for (var count27 = 0; count27 < 4500; count27++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 messakmesg gagqqpqpqp qqpflppaac ffataaaaaa aaaaaaaqsa qqqqqqqqqq 61 qqapqlrpaa dgqpsggghk sapkqvkrqr ssspelmrck rrlnfsgfgy slpqqqpaav 121 arrnerernr vklvnlgfat lrehvpngaa nkkmskvetl rsaveyiral qqlldehdav 181 saafqagvls ptispnysnd lnsmagspvs syssdegsyd plspeeqell dftnwf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count30 = 0; count30 < 1; count30++) {

for (var count29 = 0; count29 < 4500; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 messakmesg gagqqpqpqp qqpflppaac ffataaaaaa aaaaaaaqsa qqqqqqqqqq 61 qqapqlrpaa dgqpsggghk sapkqvkrqr ssspelmrck rrlnfsgfgy slpqqqpaav 121 arrnerernr vklvnlgfat lrehvpngaa nkkmskvetl rsaveyiral qqlldehdav 181 saafqagvls ptispnysnd lnsmagspvs syssdegsyd plspeeqell dftnwf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count32 = 0; count32 < 1; count32++) {

for (var count31 = 0; count31 < 4500; count31++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 messakmesg gagqqpqpqp qqpflppaac ffataaaaaa aaaaaaaqsa qqqqqqqqqq 61 qqapqlrpaa dgqpsggghk sapkqvkrqr ssspelmrck rrlnfsgfgy slpqqqpaav 121 arrnerernr vklvnlgfat lrehvpngaa nkkmskvetl rsaveyiral qqlldehdav 181 saafqagvls ptispnysnd lnsmagspvs syssdegsyd plspeeqell dftnwf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['adrenocyte 2', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function alpha\_cell() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhqdgissmn qlgglfvngr plpldtrqqi vrlavsgmrp cdisrilkvs ngcvskilgr 61 yyrtgvlepk giggskprla tppvvariaq lkgecpalfa weiqrqlcae glctqdktps 121 vssinrvlra lqedqglpct rlrspavlap avltphsgse tprgthpgtg hrnrtifsps 181 qaealekefq rgqypdsvar gklatatslp edtvrvwfsn rrakwrrqek lkwemqlpga 241 sqgltvprva pgiisaqqsp gsvptaalpa leplgpscyq lcwataperc lsdtppkacl 301 kpcwghlppq pnsldsgllc lpcpsshchl aslsgsqall wpgcpllygl e';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlgsvkmeah dlaewsyype agevyspvtp vptmaplnsy mtlnplsspy ppgglpaspl 61 psgplappap aaplgptfpg lgvsggssss gygapgpglv hgkempkgyr rplahakppy 121 syislitmai qqapgkmltl seiyqwimdl fpyyrenqqr wqnsirhsls fndcfvkvar 181 spdkpgkgsy walhpssgnm fengcylrrq krfkleekvk kggsgaattt rngtgsaast 241 ttpaatvtsp pqppppapep eaqggedvga ldcgspasst pyftglelpg elkldapynf 301 nhpfsinnlm seqtpappkl dvgfggygae ggepgvyyqg lysrsllnas';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myqslamaan hgpppgayea ggpgafmhga gaasspvyvp tprvpssvlg lsylqgggag 61 sasggasggs sggaasgagp gtqqgspgws qagadgaayt pppvsprfsf pgttgslaaa 121 aaaaaareaa ayssgggaag aglagreqyg ragfagsyss pypaymadvg aswaaaaaas 181 agpfdspvlh slpgranpaa rhpnlvdmfd dfsegrecvn cgamstplwr rdgtghylcn 241 acglyhkmng inrplikpqr rlsasrrvgl scancqtttt tlwrrnaege pvcnacglym 301 klhgvprpla mrkegiqtrk rkpknlnksk tpaapsgses lppasgassn ssnattssse 361 emrpiktepg lsshyghsss vsqtfsvsam sghgpsihpv lsalklspqg yaspvsqspq 421 tsskqdswns lvladshgdi ita';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mngeeqyyaa tqlykdpcaf qrgpapefsa sppaclymgr qpppppphpf pgalgaleqg 61 sppdispyev ppladdpava hlhhhlpaql alphppagpf pegaepgvle epnrvqlpfp 121 wmkstkahaw kgqwaggaya aepeenkrtr taytraqlle lekeflfnky isrprrvela 181 vmlnlterhi kiwfqnrrmk wkkeedkkrg ggtavggggv aepeqdcavt sgeellalpp 241 ppppggavpp aapvaaregr lppglsaspq pssvaprrpq epr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqyphpgpaa gavgvplyap tpllqpahpt pfyiedilgr gpaaptpapt lpspnssfts 61 lvspyrtpvy eptpihpafs hhsaaalaaa ygpggfggpl ypfprtvndy thallrhdpl 121 gkpllwspfl qrplhkrkgg qvrfsndqti elekkfetqk ylspperkrl akmlqlserq 181 vktwfqnrra kwrrlkqenp qsnkkeeles ldsscdqrqd lpseqnkgas ldssqcspsp 241 asqedlesei sedsdqevdi egdksyfnag';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['alpha cell', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function antennocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count34 = 0; count34 < 1; count34++) {

for (var count33 = 0; count33 < 4500; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miqeparvqa aptfeatire dnakyqetaa ggtysspncs csqrrtrpte ptkdsptsti 61 ttipividsk pnprillirv ptsnrgsnsn nnsnstrsgt ehraprtrii ripflpgsvp 121 trppsnqaap atsslvnpig sarrcristy pttgtltsrs qpetvaglpt tamvnqtvpv 181 lssapqtqtl aamagiqltg adeppsrrdv sgamptatve eidvpvfide ylqdieatss 241 tcnsengkei ftetdildfd inmfaeedli elegkktdpf spamdgmnnd nldddflqln 301 dllneeetfd mssmsgvgid inqeqeldfi nifhnsqpta mltssedpyl atysqatsts 361 sttkfiteep lladmtsfdi sytssqelgf glvedvsgsf asscsdvaaa ipapiqhpii 421 tnikrsapap vesqptkrsr lmlqiktela pafspttpei inqllnddrf eapttststs 481 tsntsissst hadivedlrs aeeetttdfs apntphsnys assscaaptc qtgyggflta 541 ptspaystas tsvfspspas gisgkrkrgr pakdhadgpd pvlmssmkse eerkayqdrl 601 knneasrvsr rktkvreeee kraedtllae nlrlrarade vasrerkfkk ylmerqrqks 661 tyvkqeqd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count36 = 0; count36 < 1; count36++) {

for (var count35 = 0; count35 < 4500; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miqeparvqa aptfeatire dnakyqetaa ggtysspncs csqrrtrpte ptkdsptsti 61 ttipividsk pnprillirv ptsnrgsnsn nnsnstrsgt ehraprtrii ripflpgsvp 121 trppsnqaap atsslvnpig sarrcristy pttgtltsrs qpetvaglpt tamvnqtvpv 181 lssapqtqtl aamagiqltg adeppsrrdv sgamptatve eidvpvfide ylqdieatss 241 tcnsengkei ftetdildfd inmfaeedli elegkktdpf spamdgmnnd nldddflqln 301 dllneeetfd mssmsgvgid inqeqeldfi nifhnsqpta mltssedpyl atysqatsts 361 sttkfiteep lladmtsfdi sytssqelgf glvedvsgsf asscsdvaaa ipapiqhpii 421 tnikrsapap vesqptkrsr lmlqiktela pafspttpei inqllnddrf eapttststs 481 tsntsissst hadivedlrs aeeetttdfs apntphsnys assscaaptc qtgyggflta 541 ptspaystas tsvfspspas gisgkrkrgr pakdhadgpd pvlmssmkse eerkayqdrl 601 knneasrvsr rktkvreeee kraedtllae nlrlrarade vasrerkfkk ylmerqrqks 661 tyvkqeqd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count38 = 0; count38 < 1; count38++) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miqeparvqa aptfeatire dnakyqetaa ggtysspncs csqrrtrpte ptkdsptsti 61 ttipividsk pnprillirv ptsnrgsnsn nnsnstrsgt ehraprtrii ripflpgsvp 121 trppsnqaap atsslvnpig sarrcristy pttgtltsrs qpetvaglpt tamvnqtvpv 181 lssapqtqtl aamagiqltg adeppsrrdv sgamptatve eidvpvfide ylqdieatss 241 tcnsengkei ftetdildfd inmfaeedli elegkktdpf spamdgmnnd nldddflqln 301 dllneeetfd mssmsgvgid inqeqeldfi nifhnsqpta mltssedpyl atysqatsts 361 sttkfiteep lladmtsfdi sytssqelgf glvedvsgsf asscsdvaaa ipapiqhpii 421 tnikrsapap vesqptkrsr lmlqiktela pafspttpei inqllnddrf eapttststs 481 tsntsissst hadivedlrs aeeetttdfs apntphsnys assscaaptc qtgyggflta 541 ptspaystas tsvfspspas gisgkrkrgr pakdhadgpd pvlmssmkse eerkayqdrl 601 knneasrvsr rktkvreeee kraedtllae nlrlrarade vasrerkfkk ylmerqrqks 661 tyvkqeqd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count40 = 0; count40 < 1; count40++) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miqeparvqa aptfeatire dnakyqetaa ggtysspncs csqrrtrpte ptkdsptsti 61 ttipividsk pnprillirv ptsnrgsnsn nnsnstrsgt ehraprtrii ripflpgsvp 121 trppsnqaap atsslvnpig sarrcristy pttgtltsrs qpetvaglpt tamvnqtvpv 181 lssapqtqtl aamagiqltg adeppsrrdv sgamptatve eidvpvfide ylqdieatss 241 tcnsengkei ftetdildfd inmfaeedli elegkktdpf spamdgmnnd nldddflqln 301 dllneeetfd mssmsgvgid inqeqeldfi nifhnsqpta mltssedpyl atysqatsts 361 sttkfiteep lladmtsfdi sytssqelgf glvedvsgsf asscsdvaaa ipapiqhpii 421 tnikrsapap vesqptkrsr lmlqiktela pafspttpei inqllnddrf eapttststs 481 tsntsissst hadivedlrs aeeetttdfs apntphsnys assscaaptc qtgyggflta 541 ptspaystas tsvfspspas gisgkrkrgr pakdhadgpd pvlmssmkse eerkayqdrl 601 knneasrvsr rktkvreeee kraedtllae nlrlrarade vasrerkfkk ylmerqrqks 661 tyvkqeqd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['antennocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function beta\_cell() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhqdgissmn qlgglfvngr plpldtrqqi vrlavsgmrp cdisrilkvs ngcvskilgr 61 yyrtgvlepk giggskprla tppvvariaq lkgecpalfa weiqrqlcae glctqdktps 121 vssinrvlra lqedqglpct rlrspavlap avltphsgse tprgthpgtg hrnrtifsps 181 qaealekefq rgqypdsvar gklatatslp edtvrvwfsn rrakwrrqek lkwemqlpga 241 sqgltvprva pgiisaqqsp gsvptaalpa leplgpscyq lcwataperc lsdtppkacl 301 kpcwghlppq pnsldsgllc lpcpsshchl aslsgsqall wpgcpllygl e';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhqdgissmn qlgglfvngr plpldtrqqi vrlavsgmrp cdisrilkvs ngcvskilgr 61 yyrtgvlepk giggskprla tppvvariaq lkgecpalfa weiqrqlcae glctqdktps 121 vssinrvlra lqedqglpct rlrspavlap avltphsgse tprgthpgtg hrnrtifsps 181 qaealekefq rgqypdsvar gklatatslp edtvrvwfsn rrakwrrqek lkwemqlpga 241 sqgltvprva pgiisaqqsp gsvptaalpa leplgpscyq lcwataperc lsdtppkacl 301 kpcwghlppq pnsldsgllc lpcpsshchl aslsgsqall wpgcpllygl e';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhqdgissmn qlgglfvngr plpldtrqqi vrlavsgmrp cdisrilkvs ngcvskilgr 61 yyrtgvlepk giggskprla tppvvariaq lkgecpalfa weiqrqlcae glctqdktps 121 vssinrvlra lqedqglpct rlrspavlap avltphsgse tprgthpgtg hrnrtifsps 181 qaealekefq rgqypdsvar gklatatslp edtvrvwfsn rrakwrrqek lkwemqlpga 241 sqgltvprva pgiisaqqsp gsvptaalpa leplgpscyq lcwataperc lsdtppkacl 301 kpcwghlppq pnsldsgllc lpcpsshchl aslsgsqall wpgcpllygl e';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhqdgissmn qlgglfvngr plpldtrqqi vrlavsgmrp cdisrilkvs ngcvskilgr 61 yyrtgvlepk giggskprla tppvvariaq lkgecpalfa weiqrqlcae glctqdktps 121 vssinrvlra lqedqglpct rlrspavlap avltphsgse tprgthpgtg hrnrtifsps 181 qaealekefq rgqypdsvar gklatatslp edtvrvwfsn rrakwrrqek lkwemqlpga 241 sqgltvprva pgiisaqqsp gsvptaalpa leplgpscyq lcwataperc lsdtppkacl 301 kpcwghlppq pnsldsgllc lpcpsshchl aslsgsqall wpgcpllygl e';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['beta cell', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function cardioblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myqslamaan hgpppgayea ggpgafmhga gaasspvyvp tprvpssvlg lsylqgggag 61 sasggasggs sggaasgagp gtqqgspgws qagadgaayt pppvsprfsf pgttgslaaa 121 aaaaaareaa ayssgggaag aglagreqyg ragfagsyss pypaymadvg aswaaaaaas 181 agpfdspvlh slpgranpaa rhpnlvdmfd dfsegrecvn cgamstplwr rdgtghylcn 241 acglyhkmng inrplikpqr rlsasrrvgl scancqtttt tlwrrnaege pvcnacglym 301 klhgvprpla mrkegiqtrk rkpknlnksk tpaapsgses lppasgassn ssnattssse 361 emrpiktepg lsshyghsss vsqtfsvsam sghgpsihpv lsalklspqg yaspvsqspq 421 tsskqdswns lvladshgdi ita';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrkkiqitr imdernrqvt ftkrkfglmk kayelsvlcd ceialiifns tnklfqyast 61 dmdkvllkyt eynephesrt nsdivetlrk kglngcdspd pdaddsvghs pesedkyrki 121 nedidlmisr qrlcavpppn fempvsipvs shnslvysnp vsslgnpnll plahpslqrn 181 smspgvthrp psagntgglm ggdltsgagt sagngygnpr nspgllvspg nlnknmqaks 241 pppmnlgmnn rkpdlrvlip pgskntmpsv sedvdlllnq rinnsqsaqs latpvvsvat 301 ptlpgqgmgg ypsaisttyg teyslssadl sslsgfntas alhlgsvtgw qqqhlhnmpp 361 salsqlgact sthlsqssnl slpstqslni ksepvspprd rtttpsrypq htrheagrsp 421 vdslsscsss ydgsdredhr nefhspiglt rpspderesp svkrmrlseg wat';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madadegfgl ahtplepdak dlpcdskpes algapsksps spqaaftqqg megikvflhe 61 relwlkfhev gtemiitkag rrmfpsykvk vtglnpktky illmdivpad dhrykfadnk 121 wsvtgkaepa mpgrlyvhpd spatgahwmr qlvsfqklkl tnnhldpfgh iilnsmhkyq 181 prlhivkade nngfgsknta fcthvfpeta fiavtsyqnh kitqlkienn pfakgfrgsd 241 dmelhrmsrm qskeypvvpr stvrqkvasn hspfssesra lstssnlgsq yqcengvsgp 301 sqdllpppnp yplpqehsqi yhctkrkeee csttdhpykk pymetspsee dsfyrssypq 361 qqglgasyrt esaqrqacmy assappsepv pslediscnt wpsmpsyssc tvttvqpmdr 421 lpyqhfsahf tsgplvprla gmanhgspql gegmfqhqts vahqpvvrqc gpqtglqspg 481 tlqppeflys hgvprtlsph qyhsvhgvgm vpewsdns';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['cardioblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function delta\_cell() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count42 = 0; count42 < 1; count42++) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtpqpsgapt vqvtreters fprasedevt cptsappspt rtrgncaeae eggcrgaprk 61 lrarrggrsr pkselalskq rrsrrkkand rernrmhnln saldalrgvl ptfpddaklt 121 kietlrfahn yiwaltqtlr iadhslyale ppaphcgelg spggspgdwg slyspvsqag 181 slspaaslee rpgllgatfs aclspgslaf sdfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count44 = 0; count44 < 1; count44++) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtpqpsgapt vqvtreters fprasedevt cptsappspt rtrgncaeae eggcrgaprk 61 lrarrggrsr pkselalskq rrsrrkkand rernrmhnln saldalrgvl ptfpddaklt 121 kietlrfahn yiwaltqtlr iadhslyale ppaphcgelg spggspgdwg slyspvsqag 181 slspaaslee rpgllgatfs aclspgslaf sdfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count46 = 0; count46 < 1; count46++) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtpqpsgapt vqvtreters fprasedevt cptsappspt rtrgncaeae eggcrgaprk 61 lrarrggrsr pkselalskq rrsrrkkand rernrmhnln saldalrgvl ptfpddaklt 121 kietlrfahn yiwaltqtlr iadhslyale ppaphcgelg spggspgdwg slyspvsqag 181 slspaaslee rpgllgatfs aclspgslaf sdfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count48 = 0; count48 < 1; count48++) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtpqpsgapt vqvtreters fprasedevt cptsappspt rtrgncaeae eggcrgaprk 61 lrarrggrsr pkselalskq rrsrrkkand rernrmhnln saldalrgvl ptfpddaklt 121 kietlrfahn yiwaltqtlr iadhslyale ppaphcgelg spggspgdwg slyspvsqag 181 slspaaslee rpgllgatfs aclspgslaf sdfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['delta cell', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function ductocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count50 = 0; count50 < 1; count50++) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtemsflsse vlvgdlmspf dqsglgaees lgllddylev akhfkphgfs sdkakagsse 61 wlavdglvsp snnskedafs gtdwmlekmd lkefdldall giddletmpd dllttlddtc 121 dlfaplvqet nkqppqtvnp ighlpesltk pdqvapftfl qplplspgvl sstpdhsfsl 181 elgsevdite gdrkpdytay vamipqcike edtpsdndsg icmspesylg spqhspstrg 241 spnrslpspg vlcgsarpkp ydppgekmva akvkgekldk klkkmeqnkt aatryrqkkr 301 aeqealtgec kelekkneal keradslake iqylkdliee vrkargkkrv p';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count52 = 0; count52 < 1; count52++) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdfdergpcs snmylpscty yvsgpdfssl psflpqtpss rpmtysyssn lpqvqpvrev 61 tfreyaiepa tkwhprgnla hcysaeelvh rdclqapsaa gvpgdvlaks sanvyhhptp 121 avssnfystv grngvlpqaf dqffetaygt penlassdyp gdksaekgpp aatatsaaaa 181 aaatgapats ssdsgggggc retaaaaeek errrrpesss spesssghte dkaggssgqr 241 trkkrcpytk yqireleref ffsvyinkek rlqlsrmlnl tdrqvkiwfq nrrmkekkin 301 rdrlqyysan pll';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count54 = 0; count54 < 1; count54++) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msfpnsspaa ntflvdslis acrsdsfyss sasmymppps admgtygmqt cgllpslakr 61 evnhqnmgmn vhpyipqvds wtdpnrscri eqpvtqqvpt csfttnikee snccmysdkr 121 nklisaevps yqrlvpescp venpevpvpg yfrlsqtyat gktqeynnsp egsstvmlql 181 nprgaakpql saaqlqmekk mnepvsgqep tkvsqvespe akgglpeers claevsvssp 241 evqekeskee iksdtptsnw ltaksgrkkr cpytkhqtle lekeflfnmy ltrerrleis 301 ksvnltdrqv kiwfqnrrmk lkkmsrenri reltanltfs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count56 = 0; count56 < 1; count56++) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msragswdmd glradgggag gapassssss vaaaaasgqc rgflsapvfa gthsgraaaa 61 aaaaaaaaaa asgfaypgts ertgssssss ssavvaarpe appakecpap tpaaaaaapp 121 sapalgygyh fgngyyscrm shgvglqqna lkssphaslg gfpvekymdv sglasssvpa 181 nevparakev sfyqgytspy qhvpgyidmv stfgsgeprh eayismegyq swtlangwns 241 qvyctkdqpq gshfwkssfp gdvalnqpdm cvyrrgrkkr vpytklqlke leneyainkf 301 inkdkrrris aatnlserqv tiwfqnrrvk dkkivsklkd tvs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['ductocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function elastin() {

peptidesequence = '1 magltaaapr pgvlllllsi lhpsrpggvp gaipggvpgg vfypgaglga lgggalgpgg 61 kplkpvpggl agaglgaglg afpavtfpga lvpggvadaa aaykaakaga glggvpgvgg 121 lgvsagavvp qpgagvkpgk vpgvglpgvy pggvlpgarf pgvgvlpgvp tgagvkpkap 181 gvggafagip gvgpfggpqp gvplgypika pklpggyglp yttgklpygy gpggvagaag 241 kagyptgtgv gpqaaaaaaa kaaakfgaga agvlpgvgga gvpgvpgaip giggiagvgt 301 paaaaaaaaa akaakygaaa glvpggpgfg pgvvgvpgag vpgvgvpgag ipvvpgagip 361 gaavpgvvsp eaaakaaaka akygarpgvg vggiptygvg aggfpgfgvg vggipgvagv 421 pgvggvpgvg gvpgvgispe aqaaaaakaa kygvgtpaaa aakaaakaaq fglvpgvgva 481 pgvgvapgvg vapgvglapg vgvapgvgva pgvgvapgig pggvaaaaks aakvaakaql 541 raaaglgagi pglgvgvgvp glgvgagvpg lgvgagvpgf gavpgalaaa kaakygaavp 601 gvlgglgalg gvgipggvvg agpaaaaaaa kaaakaaqfg lvgaaglggl gvgglgvpgv 661 gglggippaa aakaakygaa glggvlggag qfplggvaar pgfglspifp ggaclgkacg 721 rkrk';

protein();

}

function endocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhsassmlga vkmeghepsd wssyyaepeg yssvsnmnag lgmngmntym smsaaamgsg 61 sgnmsagsmn mssyvgagms pslagmspga gamagmggsa gaagvagmgp hlspslsplg 121 gqaagamggl apyanmnsms pmygqaglsr ardpktyrrs ythakppysy islitmaiqq 181 spnkmltlse iyqwimdlfp fyrqnqqrwq nsirhslsfn dcflkvprsp dkpgkgsfwt 241 lhpdsgnmfe ngcylrrqkr fkcekqlalk eaagaagsgk kaaagaqasq aqlgeaagpa 301 setpagtesp hssaspcqeh krgglgelkg tpaaalsppe papspgqqqq aaahllgpph 361 hpglppeahl kpehhyafnh pfsinnlmss eqqhhhshhh hqphkmdlka yeqvmhypgy 421 gspmpgslam gpvtnktgld asplaadtsy yqgvysrpim nss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MYVSYLLDKDVSMYPSSVRHSGGLNLAPQNFVSPPQYPDYGGYH VAAAAAAAANLDSAQSPGPSWPAAYGAPLREDWNGYAPGGAAAAANAVAHGLNGGSPA AAMGYSSPADYHPHHHPHHHPHHPAAAPSCASGLLQTLNPGPPGPAATAAAEQLSPGG QRRNLCEWMRKPAQQSLGSQVKTRTKDKYRVVYTDHQRLELEKEFHYSRYITIRRKAE LAATLGLSERQVKIWFQNRRAKERKINKKKLQQQQQQQPPQPPPPPPQPPQPQPGPLR SVPEPLSPVSSLQASVPGSVPGVLGPTGGVLNPTVTQ';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['endocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function epsilon\_cell() {

delta\_cell();

}

function epycyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count58 = 0; count58 < 1; count58++) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maghlasdfa fspppggggd gpggpepgwv dprtwlsfqg ppggpgigpg vgpgsevwgi 61 ppcpppyefc ggmaycgpqv gvglvpqggl etsqpegeag vgvesnsdga spepctvtpg 121 avklekekle qnpeesqdik alqkeleqfa kllkqkritl gytqadvglt lgvlfgkvfs 181 qtticrfeal qlsfknmckl rpllqkwvee adnnenlqei ckaetlvqar krkrtsienr 241 vrgnlenlfl qcpkptlqqi shiaqqlgle kdvvrvwfcn rrqkgkrsss dyaqredfea 301 agspfsggpv sfplapgphf gtpgygsphf talyssvpfp egeafppvsv ttlgspmhsn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count60 = 0; count60 < 1; count60++) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count62 = 0; count62 < 1; count62++) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mllrprrppp lappappspa spdpeprtpg dapgtpprrp aspsalgelg lpvspgsaqr 61 tpwsaretel llgtllqpav wrallldrrq alptyrrvsa alaqqqvrrt paqcrrrykf 121 lkdkfreahg qppgpfdeqi rklmgllgdn grkrprrrsp gsgrpqrarr pvpnahapap 181 sepdatplpt ardrdadptw tlrfspsppk sadaspapgs ppapaptala tcipedrapv 241 rgpgsppppp aredpdsppg rpedcapppa appslntall qtlghlgdia nilgplrdql 301 ltlnqhveql rgafdqtvsl avgfilgsaa aergvlrdpc q';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count64 = 0; count64 < 1; count64++) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msqqlkkrak trhqkglggr apsgakprqg kssqdlqaei epvsavwalc dgyvcyepgp 61 qalggddfsd cyiecvirge fsqpileeds lfesleylkk gseqqlsqkv feasslecsl 121 eymkkgvkke lpqkivgens leyseymtgk klppggipgi dlsdpkqlae farkkppink 181 eydslsaiac pqsgctrklr nraalrkhll ihgprdhvca ecgkafvess klkrhflvht 241 gekpfrctfe gcgkrfsldf nlrthvriht gekrfvcpfq gcnrrfiqsn nlkahiltha 301 ntnkneqegk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['epycyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function epithelioblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrqppgesdm avsdallpsf stfasgpagr ektlrqagap nnrwreelsh mkrlppvlpg 61 rpydlaaatv atdlesggag aacggsnlap lprreteefn dlldldfils nslthppesv 121 aatvsssasa ssssspsssg pasapstcsf typiragndp gvapggtggg llygresapp 181 ptapfnladi ndvspsggfv aellrpeldp vyippqqpqp pggglmgkfv lkaslsapgs 241 eygspsvisv skgspdgshp vvvapynggp prtcpkikqe avsscthlga gpplsnghrp 301 aahdfplgrq lpsrttptlg leevlssrdc hpalplppgf hphpgpnyps flpdqmqpqv 361 pplhyqgqsr gfvaragepc vcwphfgthg mmltppsspl elmppgscmp eepkpkrgrr 421 swprkrtath tcdyagcgkt ytksshlkah lrthtgekpy hcdwdgcgwk farsdeltrh 481 yrkhtghrpf qcqkcdrafs rsdhlalhmk rhf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrqppgesdm avsdallpsf stfasgpagr ektlrqagap nnrwreelsh mkrlppvlpg 61 rpydlaaatv atdlesggag aacggsnlap lprreteefn dlldldfils nslthppesv 121 aatvsssasa ssssspsssg pasapstcsf typiragndp gvapggtggg llygresapp 181 ptapfnladi ndvspsggfv aellrpeldp vyippqqpqp pggglmgkfv lkaslsapgs 241 eygspsvisv skgspdgshp vvvapynggp prtcpkikqe avsscthlga gpplsnghrp 301 aahdfplgrq lpsrttptlg leevlssrdc hpalplppgf hphpgpnyps flpdqmqpqv 361 pplhyqgqsr gfvaragepc vcwphfgthg mmltppsspl elmppgscmp eepkpkrgrr 421 swprkrtath tcdyagcgkt ytksshlkah lrthtgekpy hcdwdgcgwk farsdeltrh 481 yrkhtghrpf qcqkcdrafs rsdhlalhmk rhf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['epithelioblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function fibroblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtaasmgpvr vafvvllalc srpavgqncs gpcrcpdepa prcpagvslv ldgcgccrvc 61 akqlgelcte rdpcdphkgl fchfgspanr kigvctakdg apcifggtvy rsgesfqssc 121 kyqctcldga vgcmplcsmd vrlpspdcpf prrvklpgkc ceewvcdepk dqtvvgpala 181 ayrledtfgp dptmirancl vqttewsacs ktcgmgistr vtndnascrl ekqsrlcmvr 241 pceadleeni kkgkkcirtp kiskpikfel sgctsmktyr akfcgvctdg rcctphrttt 301 lpvefkcpdg evmkknmmfi ktcachyncp gdndifesly yrkmygdma';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['fibroblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function gamma\_cell() {

delta\_cell();

}

function genocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maapscggdr karltpslph estanpetpn stisreastq sssaatsqgy ilpegkimpn 61 tvfvggidvr mdeteirsff arygsvkevk iitdrtgvsk gygfvsffnd vdvqkivesq 121 infhgkklkl gpairkqnlc ayhvqprplv fnhppppqfq nvwtnpntet ymqptttmnp 181 itqyvqaypt ypnspvqvit gyqlpvynyq mppqwpvgeq rsyvvppays avnyhcnevd 241 pgaevvpnec svheatppsg ngpqkksvdr siqtvvsclf npenrlrnsv vtqddyfkdk 301 rvhhfrrsra mlksv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msaanpetpn stisreastq sssaaasqgw vlpegkivpn tvfvggidar mdeteigscf 61 grygsvkevk iitnrtgvsk gygfvsfvnd vdvqkivgsq ihfhgkklkl gpairkqklc 121 arhvqprplv vnpppppqfq nvwrnpntet ylqpqitpnp vtqhvqsaan petpnstisr 181 eastqsssaa asqgwvlpeg kivpntvfvg gidarmdete igscfgrygs vkevkiitnr 241 tgvskgygfv sfvndvdvqk ivgsqihfhg kklklgpair kqklcarhvq prplvvnppp 301 ppqfqnvwrn pntetylqpq itpnpvtqhv qsaanpetpn stisreastq sssaaasqgw 361 vlpegkivpn tvfvggidar mdeteigscf grygsvkevk iitnrtgvsk gygfvsfvnd 421 vdvqkivgsq ihfhgkklkl gpairkqklc arhvqprplv vnpppppqfq nvwrnpntet 481 ylqpqitpnp vtqhvqaysa yphspgqvit gcqllvynyq eyptypdsaf qvttgyqlpv 541 ynyqpfpayp rspfqvtagy qlpvynyqaf paypnspfqv atgyqfpvyn yqpfpaypss 601 pfqvtagyql pvynyqafpa ypnspfqvat gyqfpvynyq afpaypnspv qvttgyqlpv 661 ynyqafpayp sspfqvttgy qlpvynyqaf paypnsavqv ttgyqfhvyn yqmppqcpvg 721 eqrrnlwtea ykwwylvcli qrrd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 metesgpqts nqmqtdslsp spnpvspvpl nnptsapryg tvipnrifvg gidfktnesd 61 lrkffsqygs vkevkivndr agvskgygfv tfetqedaqk ilqeaeklny kdkklnigpa 121 irkqqvgipr ssimpaagtm ylttstgypy tyhngvayfh tpevtsvppp wpsrsvcssp 181 vmvaqpiyqq payhyqattq ylpgqwqwsv pqpsassapf lylqpseviy qpveiaqdgg 241 cvppplslme tsvpepysdh gvqatyhqvy apsaitmpap vmqpepiktv wsihy';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mipgnrmlmv vllcqvllgg ashaslipet gkkkvaeiqg haggrrsgqs hellrdfeat 61 llqmfglrrr pqpsksavip dymrdlyrlq sgeeeeeqih stgleyperp asrantvrsf 121 hheehlenip gtsensafrf lfnlssipen evissaelrl freqvdqgpd wergfhrini 181 yevmkppaev vpghlitrll dtrlvhhnvt rwetfdvspa vlrwtrekqp nyglaievth 241 lhqtrthqgq hvrisrslpq gsgnwaqlrp llvtfghdgr ghaltrrrra krspkhhsqr 301 arkknkncrr hslyvdfsdv gwndwivapp gyqafychgd cpfpladhln stnhaivqtl 361 vnsvnssipk accvptelsa ismlyldeyd kvvlknyqem vvegcgcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mregrgggre grsaepgpea rshsvvpsra thccsfpepf qqvcsrlavk nhglllyalf 61 svillggash aslipetgkk kvaeiqghag grrsgqshel lrdfeatllq mfglrrrpqp 121 sksavipdym rdlyrlqsge eeeeqihstg leyperpasr antvrsfhhe ehlenipgts 181 ensafrflfn lssipenevi ssaelrlfre qvdqgpdwer gfhriniyev mkppaevvpg 241 hlitrlldtr lvhhnvtrwe tfdvspavlr wtrekqpnyg laievthlhq trthqgqhvr 301 isrslpqgsg nwaqlrpllv tfghdgrgha ltrrrrakrs pkhhsqrark knkncrrhsl 361 yvdfsdvgwn dwivappgyq afychgdcpf pladhlnstn haivqtlvns vnssipkacc 421 vptelsaism lyldeydkvv lknyqemvve gcgcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhvrslraaa phsfvalwap lfllrsalad fsldnevhss fihrrlrsqe rremqreils 61 ilglphrprp hlqgkhnsap mfmldlynam aveegggpgg qgfsypykav fstqgpplas 121 lqdshfltda dmvmsfvnlv ehdkeffhpr yhhrefrfdl skipegeavt aaefriykdy 181 irerfdnetf risvyqvlqe hlgresdlfl ldsrtlwase egwlvfdita tsnhwvvnpr 241 hnlglqlsve tldgqsinpk lagligrhgp qnkqpfmvaf fkatevhfrs irstgskqrs 301 qnrsktpknq ealrmanvae nsssdqrqac kkhelyvsfr dlgwqdwiia pegyaayyce 361 gecafplnsy mnatnhaivq tlvhfinpet vpkpccaptq lnaisvlyfd dssnvilkky 421 rnmvvracgc h';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtalpgplwl lglalcalgg ggpglrpppg cpqrrlgare rrdvqreila vlglpgrprp 61 rappaasrlp asaplfmldl yhamagddde dgapaerrlg radlvmsfvn mverdralgh 121 qephwkefrf dltqipagea vtaaefriyk vpsihllnrt lhvsmfqvvq eqsnresdlf 181 fldlqtlrag degwlvldvt aasdcwllkr hkdlglrlyv etedghsvdp glagllgqra 241 prsqqpfvvt ffraspspir tpravrplrr rqpkksnelp qanrlpgifd dvhgshgrqv 301 crrhelyvsf qdlgwldwvi apqgysayyc egecsfplds cmnatnhail qslvhlmmpd 361 avpkaccapt klsatsvlyy dssnnvilrk hrnmvvkacg ch';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['genocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function glial\_cell\_1() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeeqdlsev elspvgseep rclspgsaps lgpdgggggs glraspgpge lgkvkkeqqd 61 geadddkfpv cireavsqvl sgydwtlvpm pvrvngasks kphvkrpmna fmvwaqaarr 121 kladqyphlh naelsktlgk lwrllnesdk rpfieeaerl rmqhkkdhpd ykyqprrrkn 181 gkaaqgeaec pggeaeqggt aaiqahyksa hldhrhpgeg spmsdgnpeh psgqshgppt 241 ppttpktelq sgkadpkrdg rsmgeggkph idfgnvdige ishevmsnme tfdvaeldqy 301 lppnghpghv ssysaagygl gsalavasgh sawiskppgv alptvsppgv dakaqvktet 361 agpqgpphyt dqpstsqiay tslslphygs afpsisrpqf dysdhqpsgp yyghsgqasg 421 lysafsymgp sqrplytais dpspsgpqsh spthweqpvy ttlsrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeeqdlsev elspvgseep rclspgsaps lgpdgggggs glraspgpge lgkvkkeqqd 61 geadddkfpv cireavsqvl sgydwtlvpm pvrvngasks kphvkrpmna fmvwaqaarr 121 kladqyphlh naelsktlgk lwrllnesdk rpfieeaerl rmqhkkdhpd ykyqprrrkn 181 gkaaqgeaec pggeaeqggt aaiqahyksa hldhrhpgeg spmsdgnpeh psgqshgppt 241 ppttpktelq sgkadpkrdg rsmgeggkph idfgnvdige ishevmsnme tfdvaeldqy 301 lppnghpghv ssysaagygl gsalavasgh sawiskppgv alptvsppgv dakaqvktet 361 agpqgpphyt dqpstsqiay tslslphygs afpsisrpqf dysdhqpsgp yyghsgqasg 421 lysafsymgp sqrplytais dpspsgpqsh spthweqpvy ttlsrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmtakavdki pvtlsgfvhq lsdniypved laatsvtifp naelggpfdq mngvagdgmi 61 nidmtgekrs ldlpypssfa pvsaprnqtf tymgkfsidp qypgascype giinivsagi 121 lqgvtspast tasssvtsas pnplatgplg vctmsqtqpd ldhlyspppp pppysgcagd 181 lyqdpsafls aattstsssl ayppppsyps pkpatdpglf pmipdypgff psqcqrdlhg 241 tagpdrkpfp cpldtlrvpp pltplstirn ftlggpsagv tgpgasggse gprlpgsssa 301 aaaaaaaaay nphhlplrpi lrprkypnrp sktpvherpy pcpaegcdrr fsrsdeltrh 361 irihtghkpf qcricmrnfs rsdhltthir thtgekpfac dycgrkfars derkrhtkih 421 lrqkerkssa psasvpapst ascsggvqpg gtlcssnsss lgggplapcs srtrtp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmtakavdki pvtlsgfvhq lsdniypved laatsvtifp naelggpfdq mngvagdgmi 61 nidmtgekrs ldlpypssfa pvsaprnqtf tymgkfsidp qypgascype giinivsagi 121 lqgvtspast tasssvtsas pnplatgplg vctmsqtqpd ldhlyspppp pppysgcagd 181 lyqdpsafls aattstsssl ayppppsyps pkpatdpglf pmipdypgff psqcqrdlhg 241 tagpdrkpfp cpldtlrvpp pltplstirn ftlggpsagv tgpgasggse gprlpgsssa 301 aaaaaaaaay nphhlplrpi lrprkypnrp sktpvherpy pcpaegcdrr fsrsdeltrh 361 irihtghkpf qcricmrnfs rsdhltthir thtgekpfac dycgrkfars derkrhtkih 421 lrqkerkssa psasvpapst ascsggvqpg gtlcssnsss lgggplapcs srtrtp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['glial cell 1', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function hematocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mamssflins nyvdpkfppc eeysqsdylp sdhspgyyag gqrressfqp eagfgrraac 61 tvqryaacrd pgpppppppp pppppppgls prapapppag allpepgqrc eavssspppp 121 pcaqnplhps pshsackepv vypwmrkvhv stvnpnyagg epkrsrtayt rqqvleleke 181 fhynryltrr rrveiahalc lserqikiwf qnrrmkwkkd hklpntkirs ggaagsaggp 241 pgrpnggpra l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mamssflins nyvdpkfppc eeysqsdylp sdhspgyyag gqrressfqp eagfgrraac 61 tvqryaacrd pgpppppppp pppppppgls prapapppag allpepgqrc eavssspppp 121 pcaqnplhps pshsackepv vypwmrkvhv stvnpnyagg epkrsrtayt rqqvleleke 181 fhynryltrr rrveiahalc lserqikiwf qnrrmkwkkd hklpntkirs ggaagsaggp 241 pgrpnggpra l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mamssflins nyvdpkfppc eeysqsdylp sdhspgyyag gqrressfqp eagfgrraac 61 tvqryaacrd pgpppppppp pppppppgls prapapppag allpepgqrc eavssspppp 121 pcaqnplhps pshsackepv vypwmrkvhv stvnpnyagg epkrsrtayt rqqvleleke 181 fhynryltrr rrveiahalc lserqikiwf qnrrmkwkkd hklpntkirs ggaagsaggp 241 pgrpnggpra l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mamssflins nyvdpkfppc eeysqsdylp sdhspgyyag gqrressfqp eagfgrraac 61 tvqryaacrd pgpppppppp pppppppgls prapapppag allpepgqrc eavssspppp 121 pcaqnplhps pshsackepv vypwmrkvhv stvnpnyagg epkrsrtayt rqqvleleke 181 fhynryltrr rrveiahalc lserqikiwf qnrrmkwkkd hklpntkirs ggaagsaggp 241 pgrpnggpra l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['hematocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function hepatocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 marrsafpaa alwlwsillc llalraeagp pqeeslylwi dahqarvlig feedilivse 61 gkmapfthdf rkaqqrmpai pvnihsmnft wqaagqaeyf yeflslrsld kgimadptvn 121 vpllgtvphk asvvqvgfpc lgkqdgvaaf evdvivmnse gntilktpqn aiffktcqqa 181 ecpggcrngg fcnerricec pdgfhgphce kalctprcmn gglcvtpgfc icppgfygvn 241 cdkancsttc fnggtcfypg kcicppgleg eqceiskcpq pcrnggkcig kskckcskgy 301 qgdlcskpvc epgcgahgtc hepnkcqcqe gwhgrhcnkr yeaslihalr pagaqlrqht 361 pslkkaeerr dppesnyiw';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 marrsafpaa alwlwsillc llalraeagp pqeeslylwi dahqarvlig feedilivse 61 gkmapfthdf rkaqqrmpai pvnihsmnft wqaagqaeyf yeflslrsld kgimadptvn 121 vpllgtvphk asvvqvgfpc lgkqdgvaaf evdvivmnse gntilqtpqn aiffktcqqa 181 ecpggcrngg fcnerricec pdgfhgphce kalctprcmn gglcvtpgfc icppgfygvn 241 cdkancsttc fnggtcfypg kcicppgleg eqceiskcpq pcrnggkcig kskckcskgy 301 qgdlcskpvc epgcgahgtc hepnkcqcqe gwhgrhcnkr yeaslihalr pagaqlrqht 361 pslkkaeerr dppesnyiw';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmalgaagat rvfvamvaaa lgghpllgvs atlnsvlnsn aiknlppplg gaaghpgsav 61 saapgilypg gnkyqtidny qpypcaedee cgtdeycasp trggdagvqi clacrkrrkr 121 cmrhamccpg nyckngicvs sdqnhfrgei eetitesfgn dhstldgysr rttlsskmyh 181 tkgqegsvcl rssdcasglc carhfwskic kpvlkegqvc tkhrrkgshg leifqrcycg 241 eglscriqkd hhqasnssrl htcqrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmalgaagat rvfvamvaaa lgghpllgvs atlnsvlnsn aiknlppplg gaaghpgsav 61 saapgilypg gnkyqtidny qpypcaedee cgtdeycasp trggdagvqi clacrkrrkr 121 cmrhamccpg nyckngicvs sdqnhfrgei eetitesfgn dhstldgysr rttlsskmyh 181 tkgqegsvcl rssdcasglc carhfwskic kpvlkegqvc tkhrrkgshg leifqrcycg 241 eglscriqkd hhqasnssrl htcqrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['hepatocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function infundibulocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count66 = 0; count66 < 1; count66++) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdflllqdpa stcvpepasq htlrsgpgcl qqpeqqgvrd pggiwaklga aeasaerlqg 61 rrsrgasgse pqqmgsdvrd lnallpavps lgggggcalp vsgaaqwapv ldfappgasa 121 ygslggpapp papppppppp phsfikqeps wggaepheeq clsaftvhfs gqftgtagac 181 rygpfgpppp sqassgqarm fpnapylpsc lesqpairnq gystvtfdgt psyghtpshh 241 aaqfpnhsfk hedpmgqqgs lgeqqysvpp pvygchtptd sctgsqalll rtpyssdnly 301 qmtsqlecmt wnqmnlgatl kghstgyesd nhttpilcga qyrihthgvf rgiqdvrrvp 361 gvaptlvrsa setsekrpfm caypgcnkry fklshlqmhs rkhtgekpyq cdfkdcerrf 421 srsdqlkrhq rrhtgvkpfq cktcqrkfsr sdhlkthtrt htgekpfscr wpscqkkfar 481 sdelvrhhnm hqrnmtklql al';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count68 = 0; count68 < 1; count68++) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdflllqdpa stcvpepasq htlrsgpgcl qqpeqqgvrd pggiwaklga aeasaerlqg 61 rrsrgasgse pqqmgsdvrd lnallpavps lgggggcalp vsgaaqwapv ldfappgasa 121 ygslggpapp papppppppp phsfikqeps wggaepheeq clsaftvhfs gqftgtagac 181 rygpfgpppp sqassgqarm fpnapylpsc lesqpairnq gystvtfdgt psyghtpshh 241 aaqfpnhsfk hedpmgqqgs lgeqqysvpp pvygchtptd sctgsqalll rtpyssdnly 301 qmtsqlecmt wnqmnlgatl kghstgyesd nhttpilcga qyrihthgvf rgiqdvrrvp 361 gvaptlvrsa setsekrpfm caypgcnkry fklshlqmhs rkhtgekpyq cdfkdcerrf 421 srsdqlkrhq rrhtgvkpfq cktcqrkfsr sdhlkthtrt htgekpfscr wpscqkkfar 481 sdelvrhhnm hqrnmtklql al';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count70 = 0; count70 < 1; count70++) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msresdveaq qshgssacsq phgsvtqsqg sssqsqgiss sststmpnss qsshsssgtl 61 ssletvstqe lysipedqep edqepeeptp apwarlwalq dgfanlecvn dnywfgrdks 121 ceycfdepll krtdkyrtys kkhfrifrev gpknsyiayi edhsgngtfv ntelvgkgkr 181 rplnnnseia lslsrnkvfv ffdltvddqs vypkalrdey imsktlgsga cgevklafer 241 ktckkvaiki iskrkfaigs areadpalnv eteieilkkl nhpciikikn ffdaedyyiv 301 lelmeggelf dkvvgnkrlk eatcklyfyq mllavqylhe ngiihrdlkp envllssqee 361 dclikitdfg hskilgetsl mrtlcgtpty lapevlvsvg tagynravdc wslgvilfic 421 lsgyppfseh rtqvslkdqi tsgkynfipe vwaevsekal dlvkkllvvd pkarftteea 481 lrhpwlqded mkrkfqdlls eenestalpq vlaqpstsrk rpregeaega ettkrpavca 541 avl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count72 = 0; count72 < 1; count72++) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdflllqdpa stcvpepasq htlrsgpgcl qqpeqqgvrd pggiwaklga aeasaerlqg 61 rrsrgasgse pqqmgsdvrd lnallpavps lgggggcalp vsgaaqwapv ldfappgasa 121 ygslggpapp papppppppp phsfikqeps wggaepheeq clsaftvhfs gqftgtagac 181 rygpfgpppp sqassgqarm fpnapylpsc lesqpairnq gystvtfdgt psyghtpshh 241 aaqfpnhsfk hedpmgqqgs lgeqqysvpp pvygchtptd sctgsqalll rtpyssdnly 301 qmtsqlecmt wnqmnlgatl kghstgyesd nhttpilcga qyrihthgvf rgiqdvrrvp 361 gvaptlvrsa setsekrpfm caypgcnkry fklshlqmhs rkhtgekpyq cdfkdcerrf 421 srsdqlkrhq rrhtgvkpfq cktcqrkfsr sdhlkthtrt htgekpfscr wpscqkkfar 481 sdelvrhhnm hqrnmtklql al';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['infundibulocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function isthmocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count74 = 0; count74 < 1; count74++) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmlqhpgqvs asevsasaiv pclsppgslv fedfanltpf vkeelrfaiq nkhlchrmss 61 alesvtvsdr plgvsitkae vapeederkk rrrernkiaa akcrnkkkek teclqkesek 121 lesvnaelka qieelknekq hliymlnlhr ptcivraqng rtpedernlf iqqikegtlq 181 s';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count76 = 0; count76 < 1; count76++) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmlqhpgqvs asevsasaiv pclsppgslv fedfanltpf vkeelrfaiq nkhlchrmss 61 alesvtvsdr plgvsitkae vapeederkk rrrernkiaa akcrnkkkek teclqkesek 121 lesvnaelka qieelknekq hliymlnlhr ptcivraqng rtpedernlf iqqikegtlq 181 s';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count78 = 0; count78 < 1; count78++) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqrlvawdpa clplpppppa fksmevanfy yeadclaaay ggkaapaapp aarpgprppa 61 gelgsigdhe raidfspyle plgapqapap atatdtfeaa ppapapapas sgqhhdflsd 121 lfsddyggkn ckkpaeygyv slgrlgaakg alhpgcfapl hppppppppp aelkaepgfe 181 padckrkeea gapgggagma agfpyalray lgyqavpsgs sgslstssss sppgtpspad 241 akapptacya gaapapsqvk skakktvdkh sdeykirrer nniavrksrd kakmrnletq 301 hkvleltaen erlqkkveql srelstlrnl fkqlpeplla ssghc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count80 = 0; count80 < 1; count80++) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqrlvawdpa clplpppppa fksmevanfy yeadclaaay ggkaapaapp aarpgprppa 61 gelgsigdhe raidfspyle plgapqapap atatdtfeaa ppapapapas sgqhhdflsd 121 lfsddyggkn ckkpaeygyv slgrlgaakg alhpgcfapl hppppppppp aelkaepgfe 181 padckrkeea gapgggagma agfpyalray lgyqavpsgs sgslstssss sppgtpspad 241 akapptacya gaapapsqvk skakktvdkh sdeykirrer nniavrksrd kakmrnletq 301 hkvleltaen erlqkkveql srelstlrnl fkqlpeplla ssghc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['isthmocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function lecyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count82 = 0; count82 < 1; count82++) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 malltnllpl cclallalpa qscgpgrgpv grrryarkql vpllykqfvp gvpertlgas 61 gpaegrvarg serfrdlvpn ynpdiifkde ensgadrlmt erckervnal aiavmnmwpg 121 vrlrvtegwd edghhaqdsl hyegraldit tsdrdrnkyg llarlaveag fdwvyyesrn 181 hvhvsvkadn slavraggcf pgnatvrlws gerkglrelh rgdwvlaada sgrvvptpvl 241 lfldrdlqrr asfvavetew pprkllltpw hlvfaargpa papgdfapvf arrlragdsv 301 lapggdalrp arvarvaree avgvfaplta hgtllvndvl ascyavlesh qwahrafapl 361 rllhalgall pggavqptgm hwysrllyrl aeellg';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count84 = 0; count84 < 1; count84++) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdysydedld elcpvcgdkv sgyhyglltc esckgffkrt vqnnkhytct esqsckidkt 61 qrkrcpfcrf qkcltvgmrl eavradrmrg grnkfgpmyk rdralkqqkk aqirangfkl 121 etgppmgvpp ppppapdyvl ppslhgpepk glaagppagp lgdfgapalp mavpgahgpl 181 agylypafpg raikseypep yasppqpglp ygypepfsgg pnvpelilql lqlepdedqv 241 rarilgclqe ptksrpdqpa afgllcrmad qtfisivdwa rrcmvfkele vadqmtllqn 301 cwsellvfdh iyrqvqhgke gsillvtgqe velttvatqa gsllhslvlr aqelvlqlla 361 lqldrqefvc lkfiilfsld lkflnnhilv kdaqekanaa lldytlchyp hcgdkfqqll 421 lclvevrals mqakeylyhk hlgnemprnn lliemlqakq t';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count86 = 0; count86 < 1; count86++) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdyyrkyaai flvtlsvflh vlhsapdvqe tgfhhvaqaa lkllsssnpp tkasqsarit 61 dcpectlqen pffsqpgapi lqcmgccfsr ayptplrskk tmlvqknvts estccvaksy 121 nrvtvmggfk venhtachcs tcyyhks';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count88 = 0; count88 < 1; count88++) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mktqrdghsl grwslvllll glvmplaiia qvlsykeavl raidginqrs sdanlyrlld 61 ldprptmdgd pdtpkpvsft vketvcprtt qqspedcdfk kdglvkrcmg tvtlnqargs 121 fdiscdkdnk rfallgdffr kskekigkef krivqrikdf lrnlvprtes';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count90 = 0; count90 < 1; count90++) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrallllgfl lvslestlsi ppweapkehk ykaeehtvvl tvtgepchfp fqyhrqlyhk 61 cthkgrpgpq pwcattpnfd qdqrwgycle pkkvkdhcsk hspcqkggtc vnmpsgphcl 121 cpqhltgnhc qkekcfepql lrffhkneiw yrteqaavar cqckgpdahc qrlasqacrt 181 npclhggrcl eveghrlchc pvgytgafcd vdtkascydg rglsyrglar ttlsgapcqp 241 waseatyrnv taeqarnwgl gghafcrnpd ndirpwcfvl nrdrlsweyc dlaqcqtptq 301 aapptpvspr lhvplmpaqp appkpqpttr tppqsqtpga lpakreqpps ltrngplscg 361 qrlrkslssm trvvgglval rgahpyiaal ywghsfcags liapcwvlta ahclqdrpap 421 edltvvlgqe rrnhscepcq tlavrsyrlh eafspvsyqh dlallrlqed adgscallsp 481 yvqpvclpsg aarpsettlc qvagwghqfe gaeeyasflq eaqvpflsle rcsapdvhgs 541 silpgmlcag fleggtdacq gdsggplvce dqaaerrltl qgiiswgsgc gdrnkpgvyt 601 dvayylawir ehtvs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count92 = 0; count92 < 1; count92++) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpsgphclcp qhltgnhcqk ekcfepqllr ffhkneiwyr teqaavarcq ckgpdahcqr 61 lasqacrtnp clhggrclev eghrlchcpv gytgafcdvd tkascydgrg lsyrglartt 121 lsgapcqpwa seatyrnvta eqarnwglgg hafcrnpdnd irpwcfvlnr drlsweycdl 181 aqcqtptqaa pptpvsprlh vplmpaqpap pkpqpttrtp pqsqtpgalp akreqppslt 241 rngplscgqr lrkslssmtr vvgglvalrg ahpyiaalyw ghsfcagsli apcwvltaah 301 clqdrpaped ltvvlgqerr nhscepcqtl avrsyrlhea fspvsyqhdl allrlqedad 361 gscallspyv qpvclpsgaa rpsettlcqv agwghqfega eeyasflqea qvpflslerc 421 sapdvhgssi lpgmlcagfl eggtdacqgd sggplvcedq aaerrltlqg iiswgsgcgd 481 rnkpgvytdv ayylawireh tvs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count94 = 0; count94 < 1; count94++) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgqglwrvvr nqqlqqegys eqgyltreqs rrmaasnisn tnhrkqvqgg idiyhllkar 61 kskeqegfin lemlppelsf tilsylnatd lclascvwqd landellwqg lckstwghcs 121 iynknpplgf sfrklymqld egsltfnanp degvnyfmsk gilddspkei akfifctrtl 181 nwkklriyld errdvlddlv tlhnfrnqfl pnalreffrh ihapeergey letlitkfsh 241 rfcacnpdlm relglspdav yvlcyslill sidltsphvk nkmskrefir ntrraaqnis 301 edfvghlydn iylighvaa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count96 = 0; count96 < 1; count96++) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mvlhegvnyf mskgilddsp keiakfifct rtlnwkklri ylderrdvld dlvtlhnfrn 61 qflpnalref frhihapeer geyletlitk fshrfcacnp dlmrelglsp davyvlcysl 121 illsidltsp hvknkmskre firntrraaq nisedfvghl ydniylighv aa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['lecyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function magnocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count98 = 0; count98 < 1; count98++) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdmhckadpf samhpghggv nqlggvfvng rplpdvvrqr ivelahqgvr pcdisrqlrv 61 shgcvskilg ryyetgsikp gviggskpkv atpkvvdkia eykrqnptmf aweirdrlla 121 egicdndtvp svssinriir tkvqqpfhpt pdgagtgvta pghtivpsta sppvssasnd 181 pvgsysingi lgiprsngek rkrdevevyt dpahirgggg lhlvwtlrdv segsvpngds 241 qsgvdslrkh lradtftqqq lealdrvfer psypdvfqas ehikseqgne yslpaltpgl 301 devksslsas tnpelgsnvs gtqtypvvtg rdmasttlpg ypphvpptgq gsyptstlag 361 mvpgsefsgn pyshpqytay neawrfsnpa llsspyyysa aprgsapaaa aaaydrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count100 = 0; count100 < 1; count100++) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mevtadqprw vshhhpavln gqhpdthhpg lshsymdaaq yplpeevdvl fnidgqgnhv 61 ppyygnsvra tvqryppthh gsqvcrppll hgslpwldgg kalgshhtas pwnlspfskt 121 sihhgspgpl svyppassss lsgghasphl ftfpptppkd vspdpslstp gsagsarqde 181 keclkyqvpl pdsmklessh srgsmtalgg asssthhpit typpyvpeys sglfppssll 241 ggsptgfgck srpkarsste grecvncgat stplwrrdgt ghylcnacgl yhkmngqnrp 301 likpkrrlsa arragtscan cqtttttlwr rnangdpvcn acglyyklhn inrpltmkke 361 giqtrnrkms skskkckkvh dsledfpkns sfnpaalsrh msslshispf shsshmlttp 421 tpmhppssls fgphhpssmv tamg';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count102 = 0; count102 < 1; count102++) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgsktlpapv pihpslqltn ysflqavngl ptvpsdhlpn lygfsalhav hlhqwtlgyp 61 amhlprssfs kvpgtvsslv darfqlpafp wfphviqpkp eitaggsvpa lktkprfdfa 121 nlalaatqed paklgrgegp gspagglgal ldvtklspek kptrgrlpsk tkkefvckfc 181 grhftksynl liherthtde rpytcdichk afrrqdhlrd hryihskekp fkcqecgkgf 241 cqsrtlavhk tlhsqvkelk tskikc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count104 = 0; count104 < 1; count104++) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MHVRSLRAAAPHSFVALWAPLFLLRSALADFSLDNEVHSSFIHR RLRSQERREMQREILSILGLPHRPRPHLQGKHNSAPMFMLDLYNAMAVEEGGGPGGQG FSYPYKAVFSTQGPPLASLQDSHFLTDADMVMSFVNLVEHDKEFFHPRYHHREFRFDL SKIPEGEAVTAAEFRIYKDYIRERFDNETFRISVYQVLQEHLGRESDLFLLDSRTLWA SEEGWLVFDITATSNHWVVNPRHNLGLQLSVETLDGQSINPKLAGLIGRHGPQNKQPF MVAFFKATEVHFRSIRSTGSKQRSQNRSKTPKNQEALRMANVAENSSSDQRQACKKHE LYVSFRDLGWQDWIIAPEGYAAYYCEGECAFPLNSYMNATNHAIVQTLVHFINPETVP KPCCAPTQLNAISVLYFDDSSNVILKKYRNMVVRACGCH';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count106 = 0; count106 < 1; count106++) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MHVRSLRAAAPHSFVALWAPLFLLRSALADFSLDNEVHSSFIHR RLRSQERREMQREILSILGLPHRPRPHLQGKHNSAPMFMLDLYNAMAVEEGGGPGGQG FSYPYKAVFSTQGPPLASLQDSHFLTDADMVMSFVNLVEHDKEFFHPRYHHREFRFDL SKIPEGEAVTAAEFRIYKDYIRERFDNETFRISVYQVLQEHLGRESDLFLLDSRTLWA SEEGWLVFDITATSNHWVVNPRHNLGLQLSVETLDGQSINPKLAGLIGRHGPQNKQPF MVAFFKATEVHFRSIRSTGSKQRSQNRSKTPKNQEALRMANVAENSSSDQRQACKKHE LYVSFRDLGWQDWIIAPEGYAAYYCEGECAFPLNSYMNATNHAIVQTLVHFINPETVP KPCCAPTQLNAISVLYFDDSSNVILKKYRNMVVRACGCH';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count108 = 0; count108 < 1; count108++) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MHVRSLRAAAPHSFVALWAPLFLLRSALADFSLDNEVHSSFIHR RLRSQERREMQREILSILGLPHRPRPHLQGKHNSAPMFMLDLYNAMAVEEGGGPGGQG FSYPYKAVFSTQGPPLASLQDSHFLTDADMVMSFVNLVEHDKEFFHPRYHHREFRFDL SKIPEGEAVTAAEFRIYKDYIRERFDNETFRISVYQVLQEHLGRESDLFLLDSRTLWA SEEGWLVFDITATSNHWVVNPRHNLGLQLSVETLDGQSINPKLAGLIGRHGPQNKQPF MVAFFKATEVHFRSIRSTGSKQRSQNRSKTPKNQEALRMANVAENSSSDQRQACKKHE LYVSFRDLGWQDWIIAPEGYAAYYCEGECAFPLNSYMNATNHAIVQTLVHFINPETVP KPCCAPTQLNAISVLYFDDSSNVILKKYRNMVVRACGCH';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count110 = 0; count110 < 1; count110++) {

for (var count109 = 0; count109 < 4500; count109++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MHVRSLRAAAPHSFVALWAPLFLLRSALADFSLDNEVHSSFIHR RLRSQERREMQREILSILGLPHRPRPHLQGKHNSAPMFMLDLYNAMAVEEGGGPGGQG FSYPYKAVFSTQGPPLASLQDSHFLTDADMVMSFVNLVEHDKEFFHPRYHHREFRFDL SKIPEGEAVTAAEFRIYKDYIRERFDNETFRISVYQVLQEHLGRESDLFLLDSRTLWA SEEGWLVFDITATSNHWVVNPRHNLGLQLSVETLDGQSINPKLAGLIGRHGPQNKQPF MVAFFKATEVHFRSIRSTGSKQRSQNRSKTPKNQEALRMANVAENSSSDQRQACKKHE LYVSFRDLGWQDWIIAPEGYAAYYCEGECAFPLNSYMNATNHAIVQTLVHFINPETVP KPCCAPTQLNAISVLYFDDSSNVILKKYRNMVVRACGCH';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['magnocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function mammocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mslsmrdpvi pgtsmayhpf lphrapdfam savlghqppf fpaltlppng aaalslpgal 61 akpimdqlvg aaetgipfss lgpqahlrpl ktmepeeeve ddpkvhleak elwdqfhkrg 121 temvitksgr rmfppfkvrc sgldkkakyi llmdiiaadd crykfhnsrw mvagkadpem 181 pkrmyihpds patgeqwmsk vvtfhklklt nnisdkhgft ilnsmhkyqp rfhivrandi 241 lklpystfrt ylfpetefia vtayqndkit qlkidnnpfa kgfrdtgngr rekrkqltlq 301 smrvfderhk kengtsdess seqaafncfa qasspaastv gtsnlkdlcp segesdaeae 361 skeehgpeac daakisttts eepcrdkgsp avkahlfaae rprdsgrldk aspdsrhspa 421 tissstrglg aeerrspvre gtapakveea ralpgkeafa pltvqtdaaa ahlaqgplpg 481 lgfapglagq qffnghplfl hpsqfamgga fssmaaagmg pllatvsgas tgvsgldsta 541 masaaaaqgl sgasaatlpf hlqqhvlasq glamspfgsl fpypytymaa aaaassaaas 601 ssvhrhpfln lntmrprlry spysipvpvp dgssllttal psmaaaagpl dgkvaalaas 661 pasvavdsgs elnsrsstls sssmslspkl caekeaatse lqsiqrlvsg leakpdrsrs 721 asp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqrrlvqqws vavfllsyav pscgrsvegl srrlkravse hqllhdkgks iqdlrrrffl 61 hhliaeihta eiratsevsp nskpspntkn hpvrfgsdde gryltqetnk vetykeqplk 121 tpgkkkkgkp gkrkeqekkk rrtrsawlds gvtgsglegd hlsdtsttsl eldsrrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkwilthca safphlpgcc cccflllflv ssvpvtcqal gqdmvspeat nsssssfssp 61 ssagrhvrsy nhlqgdvrwr klfsftkyfl kiekngkvsg tkkencpysi leitsveigv 121 vavkainsny ylamnkkgkl ygskefnndc klkerieeng yntyasfnwq hngrqmyval 181 ngkgaprrgq ktrrkntsah flpmvvhs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwvtkllpal llqhvllhll llpiaipyae gqrkrrntih efkksakttl ikidpalkik 61 tkkvntadqc anrctrnkgl pftckafvfd karkqclwfp fnsmssgvkk efghefdlye 121 nkdyirncii gkgrsykgtv sitksgikcq pwssmipheh sflpssyrgk dlqenycrnp 181 rgeeggpwcf tsnpevryev cdipqcseve cmtcngesyr glmdhtesgk icqrwdhqtp 241 hrhkflpery pdkgfddnyc rnpdgqprpw cytldphtrw eycaiktcad ntmndtdvpl 301 etteciqgqg egyrgtvnti wngipcqrwd sqyphehdmt penfkckdlr enycrnpdgs 361 espwcfttdp nirvgycsqi pncdmshgqd cyrgngknym gnlsqtrsgl tcsmwdknme 421 dlhrhifwep dasklnenyc rnpdddahgp wcytgnplip wdycpisrce gdttptivnl 481 dhpviscakt kqlrvvngip trtnigwmvs lryrnkhicg gslikeswvl tarqcfpsrd 541 lkdyeawlgi hdvhgrgdek ckqvlnvsql vygpegsdlv lmklarpavl ddfvstidlp 601 nygctipekt scsvygwgyt glinydgllr vahlyimgne kcsqhhrgkv tlneseicag 661 aekigsgpce gdyggplvce qhkmrmvlgv ivpgrgcaip nrpgifvrva yyakwihkii 721 ltykvpqs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['mammocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function melanoblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MTTLAGAVPRMMRPGPGQNYPRSGFPLEVSTPLGQGRVNQLGGV FINGRPLPNHIRHKIVEMAHHGIRPCVISRQLRVSHGCVSKILCRYQETGSIRPGAIG GSKPKQVTTPDVEKKIEEYKRENPGMFSWEIRDKLLKDAVCDRNTVPSVSSISRILRS KFGKGEEEEADLERKEAEESEKKAKHSIDGILSERASAPQSDEGSDIDSEPDLPLKRK QRRSRTTFTAEQLEELERAFERTHYPDIYTREELAQRAKLTEARVQVWFSNRRARWRK QAGANQLMAFNHLIPGGFPPTAMPTLPTYQLSETSYQPTSIPQAVSDPSSTVHRPQPL PPSTVHQSTIPSNPDSSSAYCLPSTRHGFSSYTDSFVPPSGPSNPMNPTIGNGLSPQV MGLLTNHGGVPHQPQTDYALSPLTGGLEPTTTVSASCSQRLDHMKSLDSLPTSQSYCP PTYSTTGYSMDPVTGYQYGQYGQSKPWTF';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MTTLAGAVPRMMRPGPGQNYPRSGFPLEVSTPLGQGRVNQLGGV FINGRPLPNHIRHKIVEMAHHGIRPCVISRQLRVSHGCVSKILCRYQETGSIRPGAIG GSKPKQVTTPDVEKKIEEYKRENPGMFSWEIRDKLLKDAVCDRNTVPSVSSISRILRS KFGKGEEEEADLERKEAEESEKKAKHSIDGILSERGKRWRLGRRTCWVTWRASAS';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqsesgivpd fevgeefhee pktyyelksq plkssssaeh pgaskppiss ssmtsrillr 61 qqlmreqmqe qerreqqqkl qaaqfmqqrv pvsqtpainv svpttlpsat qvpmevlkvq 121 thlenptkyh iqqaqrqqvk qylsttlank hanqvlslpc pnqpgdhvmp pvpgssapns 181 pmamltlnsn cekegfykfe eqnraesecp gmnthsrasc mqmddviddi islessynee 241 ilglmdpalq mantlpvsgn lidlygnqgl pppgltisns cpanlpnikr eltesearal 301 akerqkkdnh nlierrrrfn indrikelgt lipksndpdm rwnkgtilka svdyirklqr 361 eqqrakelen rqkklehanr hlllriqele mqarahglsl ipstglcspd lvnriikqep 421 vlencsqdll qhhadltctt tldltdgtit fnnnlgtgte anqaysvptk mgskledilm 481 ddtlspvgvt dpllssvspg asktssrrss msmeetehtc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqsesgivpd fevgeefhee pktyyelksq plkssssaeh pgaskppiss ssmtsrillr 61 qqlmreqmqe qerreqqqkl qaaqfmqqrv pvsqtpainv svpttlpsat qvpmevlkvq 121 thlenptkyh iqqaqrqqvk qylsttlank hanqvlslpc pnqpgdhvmp pvpgssapns 181 pmamltlnsn cekegfykfe eqnraesecp gmnthsrasc mqmddviddi islessynee 241 ilglmdpalq mantlpvsgn lidlygnqgl pppgltisns cpanlpnikr eltesearal 301 akerqkkdnh nlierrrrfn indrikelgt lipksndpdm rwnkgtilka svdyirklqr 361 eqqrakelen rqkklehanr hlllriqele mqarahglsl ipstglcspd lvnriikqep 421 vlencsqdll qhhadltctt tldltdgtit fnnnlgtgte anqaysvptk mgskledilm 481 ddtlspvgvt dpllssvspg asktssrrss msmeetehtc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['melanoblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function myoblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mellspplrd vdltapdgsl csfattddfy ddpcfdspdl rffedldprl mhvgallkpe 61 ehshfpaavh papgaredeh vrapsghhqa grcllwacka ckrkttnadr rkaatmrerr 121 rlskvneafe tlkrctssnp nqrlpkveil rnairyiegl qallrdqdaa ppgaaaafya 181 pgplppgrgg ehysgdsdas sprsncsdgm mdysgppsga rrrncyegay yneapseprp 241 gksaavssld clssiveris tespaapall ladvpsespp rrqeaaapse gessgdptqs 301 pdaapqcpag anpnpiyqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mellspplrd vdltapdgsl csfattddfy ddpcfdspdl rffedldprl mhvgallkpe 61 ehshfpaavh papgaredeh vrapsghhqa grcllwacka ckrkttnadr rkaatmrerr 121 rlskvneafe tlkrctssnp nqrlpkveil rnairyiegl qallrdqdaa ppgaaaafya 181 pgplppgrgg ehysgdsdas sprsncsdgm mdysgppsga rrrncyegay yneapseprp 241 gksaavssld clssiveris tespaapall ladvpsespp rrqeaaapse gessgdptqs 301 pdaapqcpag anpnpiyqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mellspplrd vdltapdgsl csfattddfy ddpcfdspdl rffedldprl mhvgallkpe 61 ehshfpaavh papgaredeh vrapsghhqa grcllwacka ckrkttnadr rkaatmrerr 121 rlskvneafe tlkrctssnp nqrlpkveil rnairyiegl qallrdqdaa ppgaaaafya 181 pgplppgrgg ehysgdsdas sprsncsdgm mdysgppsga rrrncyegay yneapseprp 241 gksaavssld clssiveris tespaapall ladvpsespp rrqeaaapse gessgdptqs 301 pdaapqcpag anpnpiyqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 melyetspyf yqeprfydge nylpvhlqgf eppgyertel tlspeapgpl edkglgtpeh 61 cpgqclpwac kvckrksvsv drrraatlre krrlkkvnea fealkrstll npnqrlpkve 121 ilrsaiqyie rlqallssln qeerdlryrg gggpqpgvps ecsshsascs pewgsalefs 181 anpgdhllta dptdahnlhs ltsivdsitv edvsvafpde tmpn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['myoblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function nephrocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MDPAPGVLDPRAAPPALLGTPQAEVLEDVLREQFGPLPQLAAVC RLKRLPSGGYSSTENLQLVLERRRVANAKERERIKNLNRGFARLKALVPFLPQSRKPS KVDILKGATEYIQVLSDLLEGAKDSKKQDPDEQSYSNNSSESHTSSARQLSRNITQHI SCAFGLKNEEEGPWADGGSGEPAHACRHSVMSTTEIISPTRSLDRFPEVELLSHRLPQ V';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 metncrklvs acvqlekdks qqgknedvga edpskkkrqr rqrthftsqq lqeleatfqr 61 nrypdmstre eiavwtnlte arvrvwfknr rakwrkrern qqaelckngf gpqfnglmqp 121 yddmypgysy nnwaakglts aslstksfpf fnsmnvnpls sqsmfsppns issmsmsssm 181 vpsavtgvpg sslnslnnln nlsspslnsa vptpacpyap ptppyvyrdt cnsslaslrl 241 kakqhssfgy asvqnpasnl sacqyavdrp v';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MESSAKMESGGAGQQPQPQPQQPFLPPAACFFATAAAAAAAAAA AAAQSAQQQQQQQQQQQQAPQLRPAADGQPSGGGHKSAPKQVKRQRSSSPELMRCKRR LNFSGFGYSLPQQQPAAVARRNERERNRVKLVNLGFATLREHVPNGAANKKMSKVETL RSAVEYIRALQQLLDEHDAVSAAFQAGVLSPTISPNYSNDLNSMAGSPVSSYSSDEGS YDPLSPEEQELLDFTNWF';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlwkitdnvk yeedcedrhd gssngnprvp hlssagqhly spapplshtg vaeyqpppyf 61 pppyqqlays qsadpyshlg eayaaainpl hqpaptgsqq qawpgrqsqe gaglpshhgr 121 pagllphlsg leagavsarr dayrrsdlll phahaldaag laenlglhdm phqmdevqnv 181 ddqhlllhdq tvirkgpism tknplnlpcq kelvgavmnp tevfcsvpgr lsllsstsky 241 kvtvaevqrr lsppeclnas llggvlrrak sknggrslre kldkiglnlp agrrkaahvt 301 lltslvegea vhlardfayv ceaefpskpv aeyltrphlg grnemaarkn mllaaqqlck 361 eftellsqdr tphgtsrlap vletniqncl shfslithgf gsqaicaavs alqnyikeal 421 ividksymnp gdqspadsnk tlekmekhrk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['nephrocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function ovocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maapscggdr karltpslph estanpetpn stisreastq sssaatsqgy ilpegkimpn 61 tvfvggidvr mdeteirsff arygsvkevk iitdrtgvsk gygfvsffnd vdvqkivesq 121 infhgkklkl gpairkqnlc ayhvqprplv fnhppppqfq nvwtnpntet ymqptttmnp 181 itqyvqaypt ypnspvqvit gyqlpvynyq mppqwpvgeq rsyvvppays avnyhcnevd 241 pgaevvpnec svheatppsg ngpqkksvdr siqtvvsclf npenrlrnsv vtqddyfkdk 301 rvhhfrrsra mlksv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'METESGPQTSNQMQTDSLSPSPNPVSPVPLNNPTSAPRYGTVIP NRIFVGGIDFKTNESDLRKFFSQYGSVKEVKIVNDRAGVSKGYGFVTFETQEDAQKIL QEAEKLNYKDKKLNIGPAIRKQQVGIPRSSIMPAAGTMYLTTSTGYPYTYHNGVAYFH TPEVTSVPPPWPSRSVCSSPVMVAQPIYQQPAYHYQATTQYLPGQWQWSVPQPSASSA PFLYLQPSEVIYQPVEIAQDGGCVPPPLSLMETSVPEPYSDHGVQATYHQVYAPSAIT MPAPVMQPEPIKTVWSIHY';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkklyktyat kegipksnrs hlyntvrlft pctrhkqapg dqvtgilpsv ellfnldrit 61 tvehllksvl lyninnsvsf ssavkcvcnl mikepksssr tlgrapysft fnsqfefgkk 121 hkwiqidvts llqplvasnk rsihmsinft cmkdqlehps aqnglfnmtl vspslilyln 181 dtsaqayhsw yslhykrrps qgpdqersls aypvgeeaae dgrsshhrhr rgqetvssel 241 kkplgpasfn lseyfrqfll pqnecelhdf rlsfsqlkwd nwivaphryn pryckgdcpr 301 avghrygspv htmvqniiye kldssvprps cvpakyspls vltiepdgsi aykeyedmia 361 tkctcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mellspplrd vdltapdgsl csfattddfy ddpcfdspdl rffedldprl mhvgallkpe 61 ehshfpaavh papgaredeh vrapsghhqa grcllwacka ckrkttnadr rkaatmrerr 121 rlskvneafe tlkrctssnp nqrlpkveil rnairyiegl qallrdqdaa ppgaaaafya 181 pgplppgrgg ehysgdsdas sprsncsdgm mdysgppsga rrrncyegay yneapseprp 241 gksaavssld clssiveris tespaapall ladvpsespp rrqeaaapse gessgdptqs 301 pdaapqcpag anpnpiyqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['ovocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function parathyrocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count112 = 0; count112 < 1; count112++) {

for (var count111 = 0; count111 < 4500; count111++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mafysccwvl laltwhtsay gpdqraqkkg diilgglfpi hfgvaakdqd lksrpesvec 61 irynfrgfrw lqamifaiee insspallpn ltlgyrifdt cntvskalea tlsfvaqnki 121 dslnldefcn csehipstia vvgatgsgvs tavanllglf yipqvsyass srllsnknqf 181 ksflrtipnd ehqatamadi ieyfrwnwvg tiaadddygr pgiekfreea eerdicidfs 241 elisqysdee eiqhvveviq nstakvivvf ssgpdlepli keivrrnitg kiwlaseawa 301 sssliampqy fhvvggtigf alkagqipgf reflkkvhpr ksvhngfake fweetfnchl 361 qegakgplpv dtflrghees gdrfsnssta frplctgden issvetpyid ythlrisynv 421 ylavysiaha lqdiytclpg rglftngsca dikkveawqv lkhlrhlnft nnmgeqvtfd 481 ecgdlvgnys iinwhlsped gsivfkevgy ynvyakkger lfineekilw sgfsrepltf 541 vlsvlqvpfs ncsrdclagt rkgiiegept ccfecvecpd geysdetdas acnkcpddfw 601 snenhtscia keieflswte pfgialtlfa vlgifltafv lgvfikfrnt pivkatnrel 661 sylllfsllc cfssslffig epqdwtcrlr qpafgisfvl ciscilvktn rvllvfeaki 721 ptsfhrkwwg lnlqfllvfl ctfmqivicv iwlytappss yrnqeledei ifitchegsl 781 malgfligyt cllaaicfff afksrklpen fneakfitfs mliffivwis fipayastyg 841 kfvsavevia ilaasfglla ciffnkiyii lfkpsrntie evrcstaaha fkvaaratlr 901 rsnvsrkrss slggstgstp sssissksns edpfpqperq kqqqplaltq qeqqqqpltl 961 pqqqrsqqqp rckqkvifgs gtvtfslsfd epqknamahr nsthqnslea qkssdtltrh 1021 epllplqcge tdldltvqet glqgpvggdq rpevedpeel spalvvsssq sfvisgggst 1081 vtenvvns';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count114 = 0; count114 < 1; count114++) {

for (var count113 = 0; count113 < 4500; count113++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpaaavqeav gvcsygmqls wdindpqmpq elalfdqfre wpdgyvrfiy ssdekkaqrh 61 lsgwamrntn nhnghilkks clgvvvctqa ctlpdgsrlq lrpaicdkar lkqqkkacpn 121 chsalelipc rghsgypvtn fwrldgnaif fqakgvhdhp rpesksetea rrsaikrqma 181 sfyqpqkkri reseaeenqd ssghfsnipp lenpedfdiv tetsfpipgq pcpsfpksdv 241 ykatcdlatf qgdkmppfqk ysspriylpr ppcsyelanp gytnsspypt lykdstsipn 301 dtdwvhlntl qcnvnsyssy ersfdftnkq hgwkpalgkp slvertnhgq fqamatrpyy 361 npelpcrylt tpppgapalq tvittttkvs yqayqppamk ysdsvrevks lsscnyaped 421 tgmsvypepw gppvtvtraa spsgpppmki agdcrairpt vaiphepvss rtdeaetwdv 481 clsglgsavs ysdrvgpfft ynnedf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count116 = 0; count116 < 1; count116++) {

for (var count115 = 0; count115 < 4500; count115++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlslfpspgp gssrckdkln alaisvmnqw pgvklrvteg wdedghhsee slhyegravd 61 ittsdrdrsk ygmlarlave agfdwvyyes kahihcsvka gkerkpllrd tryrasrgcs 121 pappgrgadf sfrvdgaksl ckmyawwllp aslmtkcisi pvktcitmyl kllslcsiv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count118 = 0; count118 < 1; count118++) {

for (var count117 = 0; count117 < 4500; count117++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlllarclll vlvssllvcs glacgpgrgf gkrrhpkklt playkqfipn vaektlgasg 61 ryegkisrns erfkeltpny npdiifkdee ntgadrlmtq rckdklnala isvmnqwpgv 121 klrvtegwde dghhseeslh yegravditt sdrdrskygm larlaveagf dwvyyeskah 181 ihcsvkaens vaaksggcfp gsatvhleqg gtklvkdlsp gdrvlaaddq grllysdflt 241 fldrddgakk vfyvietrep rerllltaah llfvaphnds atgepeassg sgppsggalg 301 pralfasrvr pgqrvyvvae rdgdrrllpa avhsvtlsee aagayaplta qgtilinrvl 361 ascyavieeh swahrafapf rlahallaal apartdrggd sgggdrgggg grvaltapga 421 adapgagata gihwysqlly qigtwlldse alhplgmavk ss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['parathyrocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function proophalmocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count120 = 0; count120 < 1; count120++) {

for (var count119 = 0; count119 < 4500; count119++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mftlqptpta igtvvppwsa gtlierlpsl edmahkghsg vnqlggvfvg grplpdstrq 61 kivelahsga rpcdisrilq vsngcvskil gryyetgsir praiggskpr vataevvski 121 sqykrecpsi faweirdrll qenvctndni psvssinrvl rnlaaqkeqq stgsgsssts 181 agnsisakvs vsiggnvsnv asgsrgtlss stdlmqtatp lnssesggas nsgegseqea 241 iyeklrllnt qhaagpgple paraaplvgq spnhlgtrss hpqlvhgnhq alqqhqqqsw 301 pprhysgswy ptslseipis sapniasvta yasgpslahs lsppndiesl asighqrncp 361 vatedihlkk eldghqsdet gsgegensng gasnignted dqarlilkrk lqrnrtsftn 421 dqidslekef erthypdvfa rerlagkigl peariqvwfs nrrakwrree klrnqrrtpn 481 stgasatsss tsatasltds pnslsacssl lsgsaggpsv stinglssps tlstnvnapt 541 lgagidsses ptpiphirps ctsdndngrq sedcrrvcsp cplgvgghqn thhiqsngha 601 qghalvpais prlnfnsgsf gamysnmhht alsmsdsyga vtpipsfnhs avgplappsp 661 ipqqgdltps slypchmtlr pppmapahhh ivpgdggrpa gvglgsgqsa nlgascsgsg 721 yevlsayalp pppmasssaa dssfsaassa sanvtphhti aqescpspcs sashfgvahs 781 sgfssdpisp avssyahmsy nyassantmt pssasgtsah vapgkqqffa scfyspwv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count122 = 0; count122 < 1; count122++) {

for (var count121 = 0; count121 < 4500; count121++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mftlqptpta igtvvppwsa gtlierlpsl edmahkghsg vnqlggvfvg grplpdstrq 61 kivelahsga rpcdisrilq vsngcvskil gryyetgsir praiggskpr vataevvski 121 sqykrecpsi faweirdrll qenvctndni psvssinrvl rnlaaqkeqq stgsgsssts 181 agnsisakvs vsiggnvsnv asgsrgtlss stdlmqtatp lnssesggas nsgegseqea 241 iyeklrllnt qhaagpgple paraaplvgq spnhlgtrss hpqlvhgnhq alqqhqqqsw 301 pprhysgswy ptslseipis sapniasvta yasgpslahs lsppndiesl asighqrncp 361 vatedihlkk eldghqsdet gsgegensng gasnignted dqarlilkrk lqrnrtsftn 421 dqidslekef erthypdvfa rerlagkigl peariqvwfs nrrakwrree klrnqrrtpn 481 stgasatsss tsatasltds pnslsacssl lsgsaggpsv stinglssps tlstnvnapt 541 lgagidsses ptpiphirps ctsdndngrq sedcrrvcsp cplgvgghqn thhiqsngha 601 qghalvpais prlnfnsgsf gamysnmhht alsmsdsyga vtpipsfnhs avgplappsp 661 ipqqgdltps slypchmtlr pppmapahhh ivpgdggrpa gvglgsgqsa nlgascsgsg 721 yevlsayalp pppmasssaa dssfsaassa sanvtphhti aqescpspcs sashfgvahs 781 sgfssdpisp avssyahmsy nyassantmt pssasgtsah vapgkqqffa scfyspwv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count124 = 0; count124 < 1; count124++) {

for (var count123 = 0; count123 < 4500; count123++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mftlqptpta igtvvppwsa gtlierlpsl edmahkghsg vnqlggvfvg grplpdstrq 61 kivelahsga rpcdisrilq vsngcvskil gryyetgsir praiggskpr vataevvski 121 sqykrecpsi faweirdrll qenvctndni psvssinrvl rnlaaqkeqq stgsgsssts 181 agnsisakvs vsiggnvsnv asgsrgtlss stdlmqtatp lnssesggas nsgegseqea 241 iyeklrllnt qhaagpgple paraaplvgq spnhlgtrss hpqlvhgnhq alqqhqqqsw 301 pprhysgswy ptslseipis sapniasvta yasgpslahs lsppndiesl asighqrncp 361 vatedihlkk eldghqsdet gsgegensng gasnignted dqarlilkrk lqrnrtsftn 421 dqidslekef erthypdvfa rerlagkigl peariqvwfs nrrakwrree klrnqrrtpn 481 stgasatsss tsatasltds pnslsacssl lsgsaggpsv stinglssps tlstnvnapt 541 lgagidsses ptpiphirps ctsdndngrq sedcrrvcsp cplgvgghqn thhiqsngha 601 qghalvpais prlnfnsgsf gamysnmhht alsmsdsyga vtpipsfnhs avgplappsp 661 ipqqgdltps slypchmtlr pppmapahhh ivpgdggrpa gvglgsgqsa nlgascsgsg 721 yevlsayalp pppmasssaa dssfsaassa sanvtphhti aqescpspcs sashfgvahs 781 sgfssdpisp avssyahmsy nyassantmt pssasgtsah vapgkqqffa scfyspwv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count126 = 0; count126 < 1; count126++) {

for (var count125 = 0; count125 < 4500; count125++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mftlqptpta igtvvppwsa gtlierlpsl edmahkghsg vnqlggvfvg grplpdstrq 61 kivelahsga rpcdisrilq vsngcvskil gryyetgsir praiggskpr vataevvski 121 sqykrecpsi faweirdrll qenvctndni psvssinrvl rnlaaqkeqq stgsgsssts 181 agnsisakvs vsiggnvsnv asgsrgtlss stdlmqtatp lnssesggas nsgegseqea 241 iyeklrllnt qhaagpgple paraaplvgq spnhlgtrss hpqlvhgnhq alqqhqqqsw 301 pprhysgswy ptslseipis sapniasvta yasgpslahs lsppndiesl asighqrncp 361 vatedihlkk eldghqsdet gsgegensng gasnignted dqarlilkrk lqrnrtsftn 421 dqidslekef erthypdvfa rerlagkigl peariqvwfs nrrakwrree klrnqrrtpn 481 stgasatsss tsatasltds pnslsacssl lsgsaggpsv stinglssps tlstnvnapt 541 lgagidsses ptpiphirps ctsdndngrq sedcrrvcsp cplgvgghqn thhiqsngha 601 qghalvpais prlnfnsgsf gamysnmhht alsmsdsyga vtpipsfnhs avgplappsp 661 ipqqgdltps slypchmtlr pppmapahhh ivpgdggrpa gvglgsgqsa nlgascsgsg 721 yevlsayalp pppmasssaa dssfsaassa sanvtphhti aqescpspcs sashfgvahs 781 sgfssdpisp avssyahmsy nyassantmt pssasgtsah vapgkqqffa scfyspwv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['proophalmocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function smooth\_muscle\_cell() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atgactgatt tcttttggtg ttcagagtca atataatttt ctagcaccat ctgaaatcgg 61 ttat';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atgactgatt tcttttggtg ttcagagtca atataatttt ctagcaccat ctgaaatcgg 61 ttat';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atgactgatt tcttttggtg ttcagagtca atataatttt ctagcaccat ctgaaatcgg 61 ttat';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atgactgatt tcttttggtg ttcagagtca atataatttt ctagcaccat ctgaaatcgg 61 ttat';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['smooth muscle cell', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function surfactant() {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 mdvgskevlm esppdysaap rgrfgipccp vhlkrllivv vvvvlivvvi vgallmglhm 61 sqkhtemvle msigapeaqq rlalsehlvt tatfsigstg lvvydyqqll iaykpapgtc 121 cyimkiapes ipslealtrk vhnfqmecsl qakpavptsk lgqaegrdag sapsggdpaf 181 lgmavstlcg evplyyi';

} else {

peptidesequence = '1 maeshllqwl llllptlcgp gtaawttssl acaqgpefwc qsleqalqcr alghclqevw 61 ghvgaddlcq ecedivhiln kmakeaifqd tmrkfleqec nvlplkllmp qcnqvlddyf 121 plvidyfqnq tdsngicmhl glcksrqpep eqepgmsdpl pkplrdplpd plldklvlpv 181 lpgalqarpg phtqdlseqq fpiplpycwl cralikriqa mipkgalava vaqvcrvvpl 241 vaggicqcla erysvilldt llgrmlpqlv crlvlrcsmd dsagprsptg ewlprdsech 301 lcmsvttqag nsseqaipqa mlqacvgswl drekckqfve qhtpqlltlv prgwdahttc 361 qalgvcgtms splqcihspd l';

}

}

function sustenocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqsyasamls vfnsddyspa vqenipalrr sssflctesc nskyqcetge nskgnvqdrv 61 krpmnafivw srdqrrkmal enprmrnsei skqlgyqwkm lteaekwpff qeaqklqamh 121 rekypnykyr prrkakmlpk ncsllpadpa svlcsevqld nrlyrddctk athsrmehql 181 ghlppinaas spqqrdrysh wtkl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqsyasamls vfnsddyspa vqenipalrr sssflctesc nskyqcetge nskgnvqdrv 61 krpmnafivw srdqrrkmal enprmrnsei skqlgyqwkm lteaekwpff qeaqklqamh 121 rekypnykyr prrkakmlpk ncsllpadpa svlcsevqld nrlyrddctk athsrmehql 181 ghlppinaas spqqrdrysh wtkl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnlldpfmkm tdeqekglsg apsptmseds agspcpsgsg sdtentrpqe ntfpkgepdl 61 kkeseedkfp vcireavsqv lkgydwtlvp mpvrvngssk nkphvkrpmn afmvwaqaar 121 rkladqyphl hnaelsktlg klwrllnese krpfveeaer lrvqhkkdhp dykyqprrrk 181 svkngqaeae eateqthisp naifkalqad sphsssgmse vhspgehsgq sqgpptpptt 241 pktdvqpgka dlkregrplp eggrqppidf rdvdigelss dvisnietfd vnefdqylpp 301 nghpgvpath gqvtytgsyg isstaatpas aghvwmskqq apppppqqpp qappapqapp 361 qpqaappqqp aappqqpqah tlttlssepg qsqrthikte qlspshyseq qqhspqqiay 421 spfnlphysp syppitrsqy dytdhqnsss yyshaagqgt glystftymn paqrpmytpi 481 adtsgvpsip qthspqhweq pvytqltrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnlldpfmkm tdeqekglsg apsptmseds agspcpsgsg sdtentrpqe ntfpkgepdl 61 kkeseedkfp vcireavsqv lkgydwtlvp mpvrvngssk nkphvkrpmn afmvwaqaar 121 rkladqyphl hnaelsktlg klwrllnese krpfveeaer lrvqhkkdhp dykyqprrrk 181 svkngqaeae eateqthisp naifkalqad sphsssgmse vhspgehsgq sqgpptpptt 241 pktdvqpgka dlkregrplp eggrqppidf rdvdigelss dvisnietfd vnefdqylpp 301 nghpgvpath gqvtytgsyg isstaatpas aghvwmskqq apppppqqpp qappapqapp 361 qpqaappqqp aappqqpqah tlttlssepg qsqrthikte qlspshyseq qqhspqqiay 421 spfnlphysp syppitrsqy dytdhqnsss yyshaagqgt glystftymn paqrpmytpi 481 adtsgvpsip qthspqhweq pvytqltrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['sustenocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function urocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkwilthca safphlpgcc cccflllflv ssvpvtcqal gqdmvspeat nsssssfssp 61 ssagrhvrsy nhlqgdvrwr klfsftkyfl kiekngkvsg tkkencpysi leitsveigv 121 vavkainsny ylamnkkgkl ygskefnndc klkerieeng yntyasfnwq hngrqmyval 181 ngkgaprrgq ktrrkntsah flpmvvhs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkwilthca safphlpgcc cccflllflv ssvpvtcqal gqdmvspeat nsssssfssp 61 ssagrhvrsy nhlqgdvrwr klfsftkyfl kiekngkvsg tkkencpysi leitsveigv 121 vavkainsny ylamnkkgkl ygskefnndc klkerieeng yntyasfnwq hngrqmyval 181 ngkgaprrgq ktrrkntsah flpmvvhs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkwilthca safphlpgcc cccflllflv ssvpvtcqal gqdmvspeat nsssssfssp 61 ssagrhvrsy nhlqgdvrwr klfsftkyfl kiekngkvsg tkkencpysi leitsveigv 121 vavkainsny ylamnkkgkl ygskefnndc klkerieeng yntyasfnwq hngrqmyval 181 ngkgaprrgq ktrrkntsah flpmvvhs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 1; countcell1++) {

for (countcell2 = 0; countcell2 < 4500; countcell2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkwilthca safphlpgcc cccflllflv ssvpvtcqal gqdmvspeat nsssssfssp 61 ssagrhvrsy nhlqgdvrwr klfsftkyfl kiekngkvsg tkkencpysi leitsveigv 121 vavkainsny ylamnkkgkl ygskefnndc klkerieeng yntyasfnwq hngrqmyval 181 ngkgaprrgq ktrrkntsah flpmvvhs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['urocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function thyrocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count136 = 0; count136 < 1; count136++) {

for (var count135 = 0; count135 < 4500; count135++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mwsggsgkar gweaaaggrs spgrlsrrri msmspkhttp fsvsdilspl eesykkvgme 61 ggglgaplaa yrqgqaappt aamqqhavgh hgavtaayhm taagvpqlsh savggycngn 121 lgnmselppy qdtmrnsasg pgwyganpdp rfpaisrfmg pasgmnmsgm gglgslgdvs 181 knmaplpsap rrkrrvlfsq aqvyelerrf kqqkylsape rehlasmihl tptqvkiwfq 241 nhrykmkrqa kdkaaqqqlq qdsggggggg gtgcpqqqqa qqqsprrvav pvlvkdgkpc 301 qagapapgaa slqghaqqqa qhqaqaaqaa aaaisvgsgg aglgahpghq pgsagqspdl 361 ahhaaspaal qgqvsslshl nssgsdygtm scstllygrt w';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count138 = 0; count138 < 1; count138++) {

for (var count137 = 0; count137 < 4500; count137++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mphnsirsgh gglnqlggaf vngrplpevv rqrivdlahq gvrpcdisrq lrvshgcvsk 61 ilgryyetgs irpgviggsk pkvatpkvve kigdykrqnp tmfaweirdr llaegvcdnd 121 tvpsvssinr iirtkvqqpf nlpmdscvat kslspghtli pssavtppes pqsdslgsty 181 singllgiaq pgsdkrkmdd sdqdscrlsi dsqssssgpr khlrtdafsq hhleplecpf 241 erqhypeaya spshtkgeqg lyplpllnst lddgkatltp sntplgrnls thqtypvvad 301 phspfaikqe tpevssssst psslsssafl dlqqvgsgvp pfnafphaas vygqftgqal 361 lsgremvgpt lpgypphipt sgqgsyassa iagmvagsey sgnayghtpy ssyseawrfp 421 nssllsspyy ysstsrpsap pttatafdhl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count140 = 0; count140 < 1; count140++) {

for (var count139 = 0; count139 < 4500; count139++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrlglcvval vlswthltis srgikgkrqr risaegsqac akgcelcsev ngclkcspkl 61 fillerndir qvgvclpscp pgyfdarnpd mnkcikckie hceacfshnf ctkckeglyl 121 hkgrcypacp egssaangtm ecsspaqcem sewspwgpcs kkqqlcgfrr gseertrrvl 181 hapvgdhaac sdtketrrct vrrvpcpegq krrkggqgrr enanrnlark eskeagagsr 241 rrkgqqqqqq qgtvgpltsa gpa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count142 = 0; count142 < 1; count142++) {

for (var count141 = 0; count141 < 4500; count141++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mifrvsaegs qacakgcelc sevngclkcs pklfillern dirqvgvclp scppgyfdar 61 npdmnkcikc kiehceacfs hnfctkckeg lylhkgrcyp acpegssaan gtmecsspaq 121 cemsewspwg pcskkqqlcg frrgseertr rvlhapvgdh aacsdtketr rctvrrvpcp 181 egqkrrkggq grrenanrnl arkeskeaga gsrrrkgqqq qqqqgtvgpl tsagpa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['thyrocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function uterocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count144 = 0; count144 < 1; count144++) {

for (var count143 = 0; count143 < 4500; count143++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msarkgyllp spnypttmsc sespaansfl vdslissgrg eaggggggag ggggggyyah 61 ggvylppaad lpyglqscgl fptlggkrne aaspgsgggg gglgpgahgy gpspidlwld 121 aprscrmepp dgpppppqqq pppppqppqp apqatscsfa qnikeessyc lydsadkcpk 181 vsataaelap fprgpppdgc algtssgvpv pgyfrlsqay gtakgygsgg ggaqqlgagp 241 fpaqppgrgf dlppalasgs adaarkeral dspppptlac gsgggsqgde eahasssaae 301 elspapsess kaspekdslg nskgenaanw ltaksgrkkr cpytkhqtle lekeflfnmy 361 ltrerrleis rsvhltdrqv kiwfqnrrmk lkkmnrenri reltanfnfs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count146 = 0; count146 < 1; count146++) {

for (var count145 = 0; count145 < 4500; count145++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msarkgyllp spnypttmsc sespaansfl vdslissgrg eaggggggag ggggggyyah 61 ggvylppaad lpyglqscgl fptlggkrne aaspgsgggg gglgpgahgy gpspidlwld 121 aprscrmepp dgpppppqqq pppppqppqp apqatscsfa qnikeessyc lydsadkcpk 181 vsataaelap fprgpppdgc algtssgvpv pgyfrlsqay gtakgygsgg ggaqqlgagp 241 fpaqppgrgf dlppalasgs adaarkeral dspppptlac gsgggsqgde eahasssaae 301 elspapsess kaspekdslg nskgenaanw ltaksgrkkr cpytkhqtle lekeflfnmy 361 ltrerrleis rsvhltdrqv kiwfqnrrmk lkkmnrenri reltanfnfs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count148 = 0; count148 < 1; count148++) {

for (var count147 = 0; count147 < 4500; count147++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdmhckadpf samhpghggv nqlggvfvng rplpdvvrqr ivelahqgvr pcdisrqlrv 61 shgcvskilg ryyetgsikp gviggskpkv atpkvvdkia eykrqnptmf aweirdrlla 121 egicdndtvp svssinriir tkvqqpfhpt pdgagtgvta pghtivpsta sppvssasnd 181 pvgsysingi lgiprsngek rkrdevevyt dpahirgggg lhlvwtlrdv segsvpngds 241 qsgvdslrkh lradtftqqq lealdrvfer psypdvfqas ehikseqgne yslpaltpgl 301 devksslsas tnpelgsnvs gtqtypvvtg rdmasttlpg ypphvpptgq gsyptstlag 361 mvpgsefsgn pyshpqytay neawrfsnpa llsspyyysa aprgsapaaa aaaydrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count150 = 0; count150 < 1; count150++) {

for (var count149 = 0; count149 < 4500; count149++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdmhckadpf samhpghggv nqlggvfvng rplpdvvrqr ivelahqgvr pcdisrqlrv 61 shgcvskilg ryyetgsikp gviggskpkv atpkvvdkia eykrqnptmf aweirdrlla 121 egicdndtvp svssinriir tkvqqpfhpt pdgagtgvta pghtivpsta sppvssasnd 181 pvgsysingi lgiprsngek rkrdevevyt dpahirgggg lhlvwtlrdv segsvpngds 241 qsgvdslrkh lradtftqqq lealdrvfer psypdvfqas ehikseqgne yslpaltpgl 301 devksslsas tnpelgsnvs gtqtypvvtg rdmasttlpg ypphvpptgq gsyptstlag 361 mvpgsefsgn pyshpqytay neawrfsnpa llsspyyysa aprgsapaaa aaaydrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['uterocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function vagocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count152 = 0; count152 < 1; count152++) {

for (var count151 = 0; count151 < 4500; count151++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count154 = 0; count154 < 1; count154++) {

for (var count153 = 0; count153 < 4500; count153++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count156 = 0; count156 < 1; count156++) {

for (var count155 = 0; count155 < 4500; count155++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msmlpsfgft qeqvacvcev lqqggnlerl grflwslpac dhlhknesvl kakavvafhr 61 gnfrelykil eshqfsphnh pklqqlwlka hyveaeklrg rplgavgkyr vrrkfplprt 121 iwdgeetsyc fkeksrgvlr ewyahnpyps prekrelaea tgltttqvsn wfknrrqrdr 181 aaeakerent ennnsssnkq nqlspleggk plmssseeef sppqspdqns vlllqgnmgh 241 arssnyslpg ltasqpshgl qthqhqlqds llgpltsslv dlgs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count158 = 0; count158 < 1; count158++) {

for (var count157 = 0; count157 < 4500; count157++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msmlpsfgft qeqvacvcev lqqggnlerl grflwslpac dhlhknesvl kakavvafhr 61 gnfrelykil eshqfsphnh pklqqlwlka hyveaeklrg rplgavgkyr vrrkfplprt 121 iwdgeetsyc fkeksrgvlr ewyahnpyps prekrelaea tgltttqvsn wfknrrqrdr 181 aaeakerent ennnsssnkq nqlspleggk plmssseeef sppqspdqns vlllqgnmgh 241 arssnyslpg ltasqpshgl qthqhqlqds llgpltsslv dlgs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['vagocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function vital\_components() {

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhgggppsgd sacplrtikr vqfgvlspde lkrmsvtegg ikypettegg rpklgglmdp 61 rqgviertgr cqtcagnmte cpghfghiel akpvfhvgfl vktmkvlrcv cffcskllvd 121 snnpkikdil akskgqpkkr lthvydlckg kniceggeem dnkfgveqpe gdedltkekg 181 hggcgryqpr irrsglelya ewkhvnedsq ekkillsper vheifkrisd eecfvlgmep 241 ryarpewmiv tvlpvpplsv rpavvmqgsa rnqddlthkl adivkinnql rrneqngaaa 301 hviaedvkll qfhvatmvdn elpglpramq ksgrplkslk qrlkgkegrv rgnlmgkrvd 361 fsartvitpd pnlsidqvgv prsiaanmtf aeivtpfnid rlqelvrrgn sqypgakyii 421 rdngdridlr fhpkpsdlhl qtgykverhm cdgdivifnr qptlhkmsmm ghrvrilpws 481 tfrlnlsvtt pynadfdgde mnlhlpqsle traeiqelam vprmivtpqs nrpvmgivqd 541 tltavrkftk rdvflergev mnllmflstw dgkvpqpail kprplwtgkq ifsliipghi 601 ncirthsthp ddedsgpykh ispgdtkvvv engelimgil ckkslgtsag slvhisylem 661 ghditrlfys niqtvinnwl lieghtigig dsiadsktyq diqntikkak qdvievieka 721 hnneleptpg ntlrqtfenq vnrilndard ktgssaqksl seynnfksmv vsgakgskin 781 isqviavvgq qnvegkripf gfkhrtlphf ikddygpesr gfvensylag ltptefffha 841 mggreglidt avktaetgyi qrrliksmes vmvkydatvr nsinqvvqlr ygedglages 901 vefqnlatlk psnkafekkf rfdytneral rrtlqedlvk dvlsnahiqn elerefermr 961 edrevlrvif ptgdskvvlp cnllrmiwna qkifhinprl psdlhpikvv egvkelskkl 1021 vivngddpls rqaqenatll fnihlrstlc srrmaeefrl sgeafdwllg eieskfnqai 1081 ahpgemvgal aaqslgepat qmtlntfhya gvsaknvtlg vprlkelini skkpktpslt 1141 vfllgqsard aerakdilcr lehttlrkvt antaiyydpn pqstvvaedq ewvnvyyemp 1201 dfdvarispw llrveldrkh mtdrkltmeq iaekinagfg ddlncifndd naeklvlrir 1261 imnsdenkmq eeeevvdkmd ddvflrcies nmltdmtlqg ieqiskvymh lpqtdnkkki 1321 iitedgefka lqewiletdg vslmrvlsek dvdpvrttsn diveiftvlg ieavrkaler 1381 elyhvisfdg syvnyrhlal lcdtmtcrgh lmaitrhgvn rqdtgplmkc sfeetvdvlm 1441 eaaahgesdp mkgvseniml gqlapagtgc fdllldaekc kygmeiptni pglgaagptg 1501 mffgsapspm ggispamtpw nqgatpayga wspsvgsgmt pgaagfspsa asdasgfspg 1561 yspawsptpg spgspgpssp yipspggams psysptspay eprspggytp qspsysptsp 1621 sysptspsys ptspnyspts psysptspsy sptspsyspt spsysptsps ysptspsysp 1681 tspsysptsp sysptspsys ptspsyspts psysptspsy sptspsyspt spsysptsps 1741 ysptspnysp tspnytptsp sysptspsys ptspnytpts pnysptspsy sptspsyspt 1801 spsyspsspr ytpqsptytp sspsyspssp sysptspkyt ptspsyspss peytptspky 1861 sptspkyspt spkysptspt yspttpkysp tsptysptsp vytptspkys ptsptyspts 1921 pkysptspty sptspkgsty sptspgyspt sptysltspa ispddsdeen';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MYDADEDMQYDEDDDEITPDLWQEACWIVISSYFDEKGLVRQQL DSFDEFIQMSVQRIVEDAPPIDLQAEAQHASGEVEEPPRYLLKFEQIYLSKPTHWERD GAPSPMMPNEARLRNLTYSAPLYVDITKTVIKEGEEQLQTQHQKTFIGKIPIMLRSTY CLLNGLTDRDLCELNECPLDPGGYFIINGSEKVLIAQEKMATNTVYVFAKKDSKYAYT GECRSCLENSSRPTSTIWVSMLARGGQGAKKSAIGQRIVATLPYIKQEVPIIIVFRAL GFVSDRDILEHIIYDFEDPEMMEMVKPSLDEAFVIQEQNVALNFIGSRGAKPGVTKEK RIKYAKEVLQKEMLPHVGVSDFCETKKAYFLGYMVHRLLLAALGRRELDDRDHYGNKR LDLAGPLLAFLFRGMFKNLLKEVRIYAQKFIDRGKDFNLELAIKTRIISDGLKYSLAT GNWGDQKKAHQARAGVSQVLNRLTFASTLSHLRRLNSPIGRDGKLAKPRQLHNTLWGM VCPAETPEGHAVGLVKNLALMAYISVGSQPSPILEFLEEWSMENLEEISPAAIADATK IFVNGCWVGIHKDPEQLMNTLRKLRRQMDIIVSEVSMIRDIREREIRIYTDAGRICRP LLIVEKQKLLLKKRHIDQLKEREYNNYSWQDLVASGVVEYIDTLEEETVMLAMTPDDL QEKEVAYCSTYTHCEIHPSMILGVCASIIPFPDHNQSPRNTYQSAMGKQAMGVYITNF HVRMDTLAHVLYYPQKPLVTTRSMEYLRFRELPAGINSIVAIASYTGYNQEDSVIMNR SAVDRGFFRSVFYRSYKEQESKKGFDQEEVFEKPTRETCQGMRHAIYDKLDDDGLIAP GVRVSGDDVIIGKTVTLPENEDELESTNRRYTKRDCSTFLRTSETGIVDQVMVTLNQE GYKFCKIRVRSVRIPQIGDKFASRHGQKGTCGIQYRQEDMPFTCEGITPDIIINPHAI PSRMTIGHLIECLQGKVSANKGEIGDATPFNDAVNVQKISNLLSDYGYHLRGNEVLYN GFTGRKITSQIFIGPTYYQRLKHMVDDKIHSRARGPIQILNRQPMEGRSRDGGLRFGE MERDCQIAHGAAQFLRERLFEASDPYQVHVCNLCGIMAIANTRTHTYECRGCRNKTQI SLVRMPYACKLLFQELMSMSIAPRMMSV';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MPYANQPTVRITELTDENVKFIIENTDLAVANSIRRVFIAEVPI IAIDWVQIDANSSVLHDEFIAHRLGLIPLISDDIVDKLQYSRDCTCEEFCPECSVEFT LDVRCNEDQTRHVTSRDLISNSPRVIPVTSRNRDNDPNDYVEQDDILIVKLRKGQELR LRAYAKKGFGKEHAKWNPTAGVAFEYDPDNALRHTVYPKPEEWPKSEYSELDEDESQA PYDPNGKPERFYYNVESCGSLRPETIVLSALSGLKKKLSDLQTQLSHEIQSDVLTIN';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaggsdpra gdveedasql ifpkefetae tllnsevhml lehrkqqnes aedeqelsev 61 fmktlnytar fsrfknreti asvrslllqk klhkfelacl anlcpetaee skalipsleg 121 rfedeelqqi lddiqtkrsf qy';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mddeeetyrl wkirktimql chdrgylvtq deldqtleef kaqsgdkpse grprrtdltv 61 lvahnddptd qmfvffpeep kvgiktikvy cqrmqeenit ralivvqqgm tpsakqslvd 121 mapkyileqf lqqellinit ehelvpehvv mtkeevtell aryklrenql priqagdpva 181 ryfgikrgqv vkiirpseta gryityrlvq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSDNEDNFDGDDFDDVEEDEGLDDLENAEEEGQENVEILPSGER PQANQKRITTPYMTKYERARVLGTRALQIAMCAPVMVELEGETDPLLIAMKELKARKI PIIIRRYLPDGSYEDWGVDELIITD';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfyhislehe illhpryfgp nllntvkqkl ftevegtctg kygfviavtt idnigagviq 61 pgrgfvlypv kykaivfrpf kgevvdavvt qvnkvglfte igpmscfisr hsipsemefd 121 pnsnppcykt mdediviqqd deirlkivgt rvdkndifai gslmddylgl vs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 magilfedif dvkdidpegk kfdrvsrlhc esesfkmdli ldvniqiypv dlgdkfrlvi 61 astlyedgtl ddgeynptdd rpsradqfey vmygkvyrie gdetsteaat rllrlraaew 121 qcsritgwgl lfqlcvrvlw gpaheaaggc qqpawirggl qslspdeeas llnla';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepdgtyepg fvgirfcqec nnmlypkedk enrillyacr ncdyqqeadn sciyvnkith 61 evdeltqiia dvsqdptlpr tedhpcqkcg hkeavffqsh saraedamrl yyvctaphcg 121 hrwte';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miipvrcftc gkivgnkwea ylgllqaeyt egdaldalgl kryccrrmll ahvdliekll 61 nyaplek';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnappafesf llfegekkit inkdtkvpna clftinkedh tlgniiksql lkdpqvlfag 61 ykvphplehk iiirvqttpd yspqeaftna itdliselsl leerfrvaik dkqegie';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdtqkdvqpp kqqpmiyicg echteneiks rdpircrecg yrimykkrtk rlvvfdar';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdqnnslppy aqglaspqga mtpgipifsp mmpygtgltp qpiqntnsls ileeqqrqqq 61 qqqqqqqqqq qqqqqqqqqq qqqqqqqqqq qqqqqavaaa avqqstsqqa tqgtsgqapq 121 lfhsqtltta plpgttplyp spmtpmtpit patpasessg ivpqlqnivs tvnlgckldl 181 ktialrarna eynpkrfaav imrireprtt alifssgkmv ctgakseeqs rlaarkyarv 241 vqklgfpakf ldfkiqnmvg scdvkfpirl eglvlthqqf ssyepelfpg liyrmikpri 301 vllifvsgkv vltgakvrae iyeafeniyp ilkgfrktt';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msdtdsdeds agggpfslag flfgningag qlegesvldd eckkhlaglg alglgslite 61 ltaneeltgt dgalvndegw vrstedavdy sdinevaede srryqqtmgs lqplchsdyd 121 eddydadced idcklmpppp pppgpmkkdk dqdsitgvse ngegiilpsi iapsslasek 181 vdfssssdse semgpqeatq aesedgkltl plagimqhda tkllpsvtel fpefrpgkvl 241 rflrlfgpgk nvpsvwrsar rkrkkkhrel iqeeqiqeve csvesevsqk slwnydyapp 301 pppeqclsdd eitmmapves kfsqstgdid kvtdtkprva ewrygparlw ydmlgvpedg 361 sgfdygfklr ktehepviks rmieefrkle enngtdllad enflmvtqlh weddiiwdge 421 dvkhkgtkpq raslagwlps smtrnamayn vqqgfaatld ddkpwysifp idnedlvygr 481 wedniiwdaq amprlleppv ltldpndenl ileipdekee atsnspskes kkesslkksr 541 illgktgvik eepqqnmsqp evkdpwnlsn deyyypkqqg lrgtfggnii qhsipavelr 601 qpffpthmgp iklrqfhrpp lkkysfgals qpgphsvqpl lkhikkkakm reqerqasgg 661 gemffmrtpq dltgkdgdli laeyseengp lmmqvgmatk iknyykrkpg kdpgapdcky 721 getvychtsp flgslhpgql lqafennlfr apiylhkmpe tdfliirtrq gyyirelvdi 781 fvvgqqcplf evpgpnskra nthirdflqv fiyrlfwksk drprrirmed ikkafpshse 841 ssirkrlklc adfkrtgmds nwwvlksdfr lpteeeiram vspeqccayy smiaaeqrlk 901 dagygeksff apeeeneedf qmkiddevrt apwnttrafi aamkgkclle vtgvadptgc 961 gegfsyvkip nkptqqkddk epqpvkktvt gtdadlrrls lknakqllrk fgvpeeeikk 1021 lsrwevidvv rtmsteqars gegpmskfar gsrfsvaehq erykeecqri fdlqnkvlss 1081 tevlstdtds ssaedsdfee mgknienmlq nkktssqlsr ereeqerkel qrmllaagsa 1141 asgnnhrddd tasvtslnss atgrclkiyr tfrdeegkey vrcetvrkpa vidayvrirt 1201 tkdeefirkf alfdeqhree mrkerrriqe qlrrlkrnqe keklkgppek kpkkmkerpd 1261 lklkcgacga ighmrtnkfc plyyqtnapp snpvamteeq eeelektvih ndneelikve 1321 gtkivlgkql iesadevrrk slvlkfpkqq lppkkkrrvg ttvhcdylnr phksihrrrt 1381 dpmvtlssil esiindmrdl pntypfhtpv nakvvkdyyk iitrpmdlqt lrenvrkrly 1441 psreefrehl elivknsaty ngpkhsltqi sqsmldlcde klkekedkla rlekainpll 1501 ddddqvafsf ildnivtqkm mavpdswpfh hpvnkkfvpd yykvivnpmd letirknisk 1561 hkyqsresfl ddvnlilans vkyngpesqy tktaqeivnv cyqtlteyde hltqlekdic 1621 takeaaleea elesldpmtp gpytpqppdl ydtntslsms rdasvfqdes nmsvldipsa 1681 tpekqvtqeg edgdgdlade eegtvqqpqa svlyedllms egeddeedag sdeegdnpfs 1741 aiqlsesgsd sdvgsggirp kqprmlqent rmdmeneesm msyegdggea shgledsnis 1801 ygsyeepdpk sntqdtsfss iggyevseee edeeeeeqrs gpsvlsqvhl sedeedsedf 1861 hsiagdsdld sde';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpltgvepar mnrkkgdkgf esprpyklth qvvcinninf qrksvvgfve ltifptvanl 61 nriklnskqc riyrvrindl eaafiyndpt levchseskq rnlnyfsnay aaavsavdpd 121 agngelcikv pselwkhvde lkvlkihinf sldqpkgglh fvvpsvegsm aergahvfsc 181 gyqnstrfwf pcvdsyselc twkleftvda amvavsngdl vetvythdmr kktfhymlti 241 ptaasnisla igpfeilvdp ymhevthfcl pqllpllkht tsylhevfef yeeiltcryp 301 yscfktvfid eayvevaaya smsifstnll hsamiidetp ltrrclaqsl aqqffgcfis 361 rmswsdewvl kgisgyiygl wmkktfgvne yrhwikeeld kivayelktg gvllhpifgg 421 gkekdnpash lhfsikhpht lsweyysmfq ckahlvmrli enrismefml qvfnkllsla 481 stassqkfqs hmwsqmlvst sgflksisnv sgkdiqplik qwvdqsgvvk fygsfafnrk 541 rnvleleikq dytspgtqky vgplkvtvqe ldgsfnhtlq ieenslkhdi pchsksrrnk 601 kkkiplmnge evdmdlsamd adspllwiri dpdmsvlrkv efeqadfmwq yqlryerdvv 661 aqqesilale kfptpasrla ltdileqeqc fyrvrmsacf clakiansmv stwtgppamk 721 slftrmfcck scpnivktnn fmsfqsyflq ktmpvamall rdvhnlcpke vltfildlik 781 yndnrknkfs dnyyraemid alansvtpav svnnevrtld nlnpdvrlil eeitrflnme 841 kllpsyrhti tvsclrairv lqknghvpsd palfksyaey ghfvdiriaa leavvdytkv 901 drsyeelqwl lnmiqndpvp yvrhkilnml tknppftknm esplcnealv dqlwklmnsg 961 tshdwrlrcg avdlyftlfg lsrpsclplp elglvlnlke kkavlnptii pesvagnqea 1021 annpsshpql vgfqnpfsss qdeeeidmdt vhdsqafish hlnmlerpst pglskyrpas 1081 srsalipqhs agcdstpttk pqwslelark gtgkeqaple msmhpaasap lsvftkesta 1141 skhsdhhhhh hhehkkkkkk hkhkhkhkhk hdskekdkep ftfsspasgr sirspslsd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mcesysrsll rvsvaqicqa lgwdsvqlsa chlltdvlqr ylqqlgrgch ryselygrtd 61 pilddvgeaf qlmgvslhel edyihniepv tfphqipsfp vsknnvlqfp qpgskdaeer 121 keyipdylpp ivssqeeeee eqvptdggts aeamqvplee ddeleeeeii ndenflgkrp 181 ldspeaeelp amkrprllst kgdtldvvll eareplssin tqkippmlsp vhvqdstdla 241 ppspeppmla pvaksqmpta kpletksftp ktktktsspg qktkspktaq spamvgspir 301 spktvskekk spgrskspks pkspkvtthi pqtpvrpetp nrtpsatlse kisketiqvk 361 qiqtppdagk lnsenqpkka vvadktieas idaviaraca erepdpfefs sgsesegdif 421 tspkrisgpe cttpkastsa nnftksgstp lplsggtsss dnswtmdasi devvrkaklg 481 tpsnmppnfp yisspsvspp tpeplhkvye ektklpssve vkkklkkelk tkmkkkekqr 541 drerekdknk dkskekdkvk ekekdketgr etkypwkefl keeeadpykf kikefedvdp 601 kvklkdglvr kekekhkdkk kdrekgkkdk dkrekekvkd kgredkmkap applvlppke 661 lalplfspat asrvpamlps llpvlpeklf eekekvkeke kkkdkkekkk kkekekekke 721 kerekekrer ekrekekekh khekikvepv alapspvipr ltlrvgagqd kiviskvvpa 781 peakpapsqn rpktpppapa papgpmlvsp apvplpllaq aaagpallps pgpaasgasa 841 kapvrsvvte tvstyvirde wgnqiwicpg cnkpddgspm igcddcddwy hwpcvgimta 901 ppeemqwfcp kcankkkdkk hkkrkhrah';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maagsdllde vffnsevdek vvsdlvgsle sqlaasaahh hhlaprtpev raaaagalgn 61 hvvsgspaga agagpaapae gapgaapepp pagrarpggg gpqrpgppsp rrplvpagpa 121 ppaaklrppp egsagscapv paaaavaagp epapagpakp agpaalaara gpgpgpgpgp 181 gpgpgpgkpa gpgaaqtlng saallnshha aapavslvnn gpaallplpk paapgtviqt 241 ppfvgaaapp apaapsppaa papaapaaap pppppapatl arppghpagp ptaapavppp 301 aaaqnggsag aapapapaag gpagvsgqpg pgaaaaapap gvkaespkrv vqaappaaqt 361 laasgpasta asmvigptmq galpspaavp ppapgtptgl pkgaagavtq slsrtptatt 421 sgiratltpt vlaprlpqpp qnptniqnfq lppgmvlvrs engqllmipq qalaqmqaqa 481 haqpqttmap rpatptsapp vqistvqapg tpiiarqvtp ttiikqvsqa qttvqpsatl 541 qrspgvqpql vlggaaqtas lgtatavqtg tpqrtvpgat ttssaatetm envkkcknfl 601 stliklassg kqstetaanv kelvqnlldg kieaedftsr lyrelnsspq pylvpflkrs 661 lpalrqltpd saafiqqsqq qpppptsqat taltavvlss svqrtagkta atvtsalqpp 721 vlsltqptqv gvgkqgqptp lviqqppkpg alirppqvtl tqtpmvalrq phnrimlttp 781 qqiqlnplqp vpvvkpavlp gtkalsavsa qaaaaqknkl kepgggsfrd dddindvasm 841 agvnlseesa rilatnselv gtltrsckde tfllqaplqr rileigkkhg itelhpdvvs 901 yvshatqqrl qnlvekiset aqqknfsykd ddryeqasdv raqlkffeql dqiekqrkde 961 qereilmraa ksrsrqedpe qlrlkqkake mqqqelaqmr qrdanltala aigprkkrkv 1021 dcpgpgsgae gsgpgsvvpg ssgvgtprqf trqritrvnl rdlifclene retshsllly 1081 kaflk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maalaeeqte vavklepegp ptllppqagd gagegsggtt nngpnggggn vaassstggd 61 ggtpkptvav saaapagaap vpaaapdaga phdrqtllav lqflrqsklr eaeealrrea 121 glleeavags gapgevdsag aevtsallsr vtasapgpaa pdppgtgasg atvvsgsasg 181 paapgkvgsv avedqpdvsa vlsaynqqgd ptmyeeyysg lkhfiecsld chraelsqlf 241 yplfvhmyle lvynqhenea ksffekfhgd qecyyqddlr vlssltkkeh mkgnetmldf 301 rtskfvlris rdsyqllkrh lqekqnnqiw nivqehlyid ifdgmprskq qidamvgsla 361 geakreanks kvffgllkep eievpldded eegeneegkp kkkkpkkdsi gskskkqdpn 421 appqnriplp elkdsdkldk imnmkettkr vrlgpdclps icfytflnay qgltavdvtd 481 dssliaggfa dstvrvwsvt pkklrsvkqa sdlslidkes ddvlerimde ktaselkily 541 ghsgpvygas fspdrnylls ssedgtvrlw slqtftclvg ykghnypvwd tqfspygyyf 601 vsgghdrvar lwatdhyqpl rifaghladv nctrfhpnsn yvatgsadrt vrlwdvlngn 661 cvriftghkg pihsltfspn grflatgatd grvllwdigh glmvgelkgh tdtvcslrfs 721 rdgeilasgs mdntvrlwda ikafedletd dfttatghin lpensqelll gtymtkstpv 781 vhlhftrrnl vlaagayspq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeekklkls ntvlpsesmk vvaesmgiaq iqeetcqllt devsyrikei aqdalkfmhm 61 gkrqklttsd idyalklknv eplygfhaqe fipfrfasgg grelyfyeek evdlsdiint 121 plprvpldvc lkahwlsieg cqpaipenpp papkeqqkae ateplksakp gqeedgplkg 181 kgqgattadg kgkekkappl legaplrlkp rsihelsveq qlyykeitea cvgsceakra 241 ealqsiatdp glyqmlprfs tfisegvrvn vvqnnlalli ylmrmvkalm dnptlyleky 301 vhelipavmt civsrqlclr pdvdnhwalr dfaarlvaqi ckhfstttnn iqsritktft 361 kswvdektpw ttrygsiagl aelghdvikt lilprlqqeg erirsvldgp vlsnidriga 421 dhvqslllkh capvlaklrp ppdnqdayra efgslgpllc sqvvkaraqa alqaqqvnrt 481 tltitqprpt ltlsqapqpg prtpgllkvp gsialpvqtl vsaraaappq psppptkfiv 541 msssssapst qqvlslstsa pgsgstttsp vtttvpsvqp ivklvstatt appstapsgp 601 gsvqkyivvs lpptgegkgg ptshpspvpp passpsplsg salcggkqea gdspppapgt 661 pkangsqpns gspqpap';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskskddaph elesqfilrl ppeyastvrr avqsghvnlk drltielhpd grhgivrvdr 61 vplasklvdl pcvmeslkti dkktfyktad icqmlvstvd gdlyppveep vastdpkask 121 kkdkdkekkf iwnhgitlpl knvrkrrfrk takkkyiesp dvekevkrll stdaeavstr 181 weiiaedetk eaenqgldis spgmsghrqg hdslehdelr eifndlssss ededetqhqd 241 eediniidte edlerqlqdk lnesdeqhqe negtnqlvmg iqkqidnmkg klqetqdrak 301 rqedlimkve nlalknrfqa vldelkqked rekeqlsslq eelesllek';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madaaataga ggsgtrsgsk qstnpadnyh larrrtlqvv vssllteagf esaekasvet 61 ltemlqsyis eigrsaksyc ehtartqptl sdivvtlvem gfnvdtlpay akrsqrmvit 121 appvtnqpvt pkaltagqnr phpphipshf pefpdphtyi ktptyrepvs dyqvlrekaa 181 sqrrdveral trfmaktget qslfkddvst fpliaarpft ipyltallps elemqqmeet 241 dsseqdeqtd tenlalhism edsgaekent svlqqnpsls gsrngeenii dnpylrpvkk 301 pkirrkksls';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mesgktaspk smpkdaqmma qilkdmgite yeprvinqml efafryvtti lddakiyssh 61 akkatvdadd vrlaiqcrad qsftsppprd flldiarqrn qtplplikpy sgprlppdry 121 cltapnyrlk slqkkastsa gritvprlsv gsvtsrpstp tlgtptpqtm svstkvgtpm 181 sltgqrftvq mptsqspavk asipatsavq nvlinpslig sknilittnm mssqntanes 241 snalkrkred dddddddddd ydnl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mscsgsgadp eaapasaasa pgpappvsap aalpsstaae nkaspagtag gpgagaaagg 61 tgplaarage paerrgaapv saggaappeg aisngvyvlp saangdvkpv vsstplvdfl 121 mqledytpti pdavtgyyln ragfeasdpr iirlislaaq kfisdianda lqhckmkgta 181 sgssrskskd rkytltmedl tpalseygin vkkphyft';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mddahespsd kggetgesde taavpgdpga tdtdgipeet dgdadvdlke aaaeegeles 61 qdvsdlttve redssllnpa akklkidtke kkekkqkvde deiqkmqilv ssfseeqlnr 121 yemyrrsafp kaaikrliqs itgtsvsqnv viamsgiskv fvgevveeal dvcekwgemp 181 plqpkhmrea vrrlkskgqi pnskhkkiif f';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnqfgpsali nlsnfssikp epastppqgs manstavvki pgtpgaggrl spennqvltk 61 kklqdlvrev dpneqldedv eemllqiadd fiesvvtaac qlarhrksst levkdvqlhl 121 erqwnmwipg fgseeirpyk kactteahkq rmalirkttk k';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madeeedptf eeeneeiggg aeggqgkrkr lfskelrcmm ygfgddqnpy tesvdiledl 61 viefitemth kamsigrqgr vqvedivfli rkdprkfarv kdlltmneel krarkafdea 121 nygs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msdsgsygqs ggeqqsysty gnpgsqgygq asqsysgygq ttdssygqny sgyssygqsq 61 sgysqsyggy enqkqssysq qpynnqgqqq nmessgsqgg rapsydqpdy gqqdsydqqs 121 gydqhqgsyd eqsnydqqhd sysqnqqsyh sqrenyshht qddrrdvsry gednrgyggs 181 qgggrgrggy dkdgrgpmtg ssggdrggfk nfgghrdygp rtdadsesdn sdnntifvqg 241 lgegvstdqv geffkqigii ktnkktgkpm inlytdkdtg kpkgeatvsf ddppsakaai 301 dwfdgkefhg niikvsfatr rpefmrgggs gggrrgrggy rgrggfqgrg gdpksgdwvc 361 pnpscgnmnf arrnscnqcn eprpedsrps ggdfrgrgyg gergyrgrgg rggdrggygg 421 drsgggyggd rssgggysgd rsgggyggdr sgggyggdrg ggyggdrggg yggdrgggyg 481 gdrggyggdr gggyggdrgg yggdrggygg drggyggdrg gyggdrsrgg yggdrgggsg 541 yggdrsggyg gdrsgggygg drgggyggdr ggyggkmggr ndyrndqrnr py';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpagltepag aappaavsas gtvtmapaga lpvrvestpv algavtkapv svcveptasq 61 plrspvgtlv tkvapvsapp kvssgprlpa pqivavkapn tttiqfpanl qlppgtvlik 121 snsgplmlvs pqqtvtraet tsnitsrpav panpqtvkic tvpnsssqli kkvavtpvkk 181 laqigttvvt tvpkpssvqs vavptsvvtv tpgkplntvt tlkpsslgas stpsnepnlk 241 aensaavqin lsptmlenvk kcknflamli klacsgsqsp emgqnvkklv eqlldakiea 301 eeftrklyve lksspqphlv pflkksvval rqllpnsqsf iqqcvqqtss dmviatcttt 361 vttspvvttt vsssqseksi ivsgataprt vsvqtlnpla gpvgakagvv tlhsvgptaa 421 tggttagtgl lqtskplvts vantvttvsl qpekpvvsgt avtlslpavt fgetsgaaic 481 lpsvkpvvss agttsdkpvi gtpvqiklaq pgpvlsqpag ipqavqvkql fslfqvvqqp 541 sggnekqvtt ishsstltiq kcgqktmpvn tiiptsqfpp asilkqitlp gnkilslqas 601 ptqknriken vtscfrdedd indvtsmagv nlneenacil atnselvgtl iqsckdepfl 661 figalqkril digkkhdite lnsdavnlis qatqerlrgl lekltaiaqh rmttykasen 721 yilcsdtrsq lkflekldql ekqrkdleer emllkaaksr snkedpeqlr lkqkakelqq 781 lelaqiqhrd anltalaaig prkkrplesg ieglkdnlla sgtssltatk qlhrpritri 841 clrdlifcme qeremkysra lylallk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mesgkmappk naprdalvma qilkdmgite yeprvinqml efafryvtti lddakiyssh 61 akkpnvdadd vrlaiqcrad qsftsppprd flldiarqkn qtplplikpy agprlppdry 121 cltapnyrlk slikkgpnqg rlvprlsvga vsskpttpti atpqtvsvpn kvatpmsvts 181 qrftvqipps qstpvkpvpa ttavqnvlin psmigpknil ittnmvssqn taneanplkr 241 kheddddndi m';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mansantntv pklyrsvied vindvrdifl ddgvdeqvlm elktlwenkl mqsravdgfh 61 seeqqlllqv qqqhqpqqqq hhhhhhhqqa qpqqtvpqqa qtqqvlipas qqatapqviv 121 pdskliqhmn asnmsaaata atlalpagvt pvqqiltnsg qllqvvraan gaqyifqpqq 181 svvlqqqvip qmqpggvqap viqqvlaplp ggispqtgvi iqpqqilftg nktqvipttv 241 aaptpaqaqi tatgqqqpqa qpaqtqaplv lqvdgtgdts seededeeed ydddeeedke 301 kdgaedgqve eeplnseddv sdeegqelfd tenvvvcqyd kihrsknkwk fhlkdgimnl 361 ngrdyifska igdaew';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mayqlyrntt lgnslqesld eliqsqqitp qlalqvllqf dkainaalaq rvrnrvnfrg 61 slntyrfcdn vwtfvlndve frevtelikv dkvkivacdg kntgsntte';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mastsrldal prvtcpnhpd ailvedyrag dmicpecglv vgdrvidvgs ewrtfsndka 61 tkdpsrvgds qnpllsdgdl stmigkgtga asfdefgnsk yqnrrtmsss drammnafke 121 ittmadrinl prnivdrtnn lfkqvyeqks lkgrandaia saclyiacrq egvprtfkei 181 cavsriskke igrcfklilk aletsvdlit tgdfmsrfcs nlclpkqvqm aathiarkav 241 eldlvpgrsp isvaaaaiym asqasaekrt qkeigdiagv advtirqsyr liyprapdlf 301 ptdfkfdtpv dklpql';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maalgpssqn vteyvvrvpk nttkkynima fnaadkvnfa twnqarlerd lsnkkiyqee 61 empesgagse fnrklreear rkkygivlke frpedqpwll rvngksgrkf kgikkggvte 121 ntsyyiftqc pdgafeafpv hnwynftpla rhrtltaeea eeewerrnkv lnhfsimqqr 181 rlkdqdqded eeekekrgrr kaselrihdl eddlemssda sdasgeeggr vpkakkkapl 241 akggrkkkkk kgsddeafed sddgdfegqe vdymsdgsss sqeepeskak apqqeegpkg 301 vdeqsdssee seeekppeed keeeeekkap tpqekkrrkd sseesdssee sdidseassa 361 lfmakkktpp krerkpsggs srgnsrpgtp saeggstsst lraaaskleq gkrvsempaa 421 krlrldtgpq slsgkstpqp psgkttpnsg dvqvtedavr ryltrkpmtt kdllkkfqtk 481 ktglsseqtv nvlaqilkrl nperkmindk mhfslke';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maergeldlt gakqntgvwl vkvpkylsqq wakasgrgev gklriaktqg rtevsftlne 61 dlanihdigg kpasvsapre hpfvlqsvgg qtltvftess sdklslegiv vqraecrpaa 121 senymrlkrl qieesskpvr lsqqldkvvt tnykpvanhq ynieyerkkk edgkraradk 181 qhvldmlfsa fekhqyynlk dlvditkqpv vylkeilkei gvqnvkgihk ntwelkpeyr 241 hyqgeeksd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (countcell1 = 0; countcell1 < 4500; countcell1++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madpdvltev paalkrlaky virgfygieh alaldilirn scvkeedmle llkfdrkqlr 61 svlnnlkgdk fikcrmrvet aadgkttrhn yyfinyrtlv nvvkykldhm rrrietderd 121 stnrasfkcp vcsstftdle anqlfdpmtg tfrctfchte veedesampk kdartllarf 181 neqiepiyal lretedvnla yeilepepte ipalkqskdh aattagaasl agghhreawa 241 tkgpsyedly tqnvvinmdd qedlhrasle gksakerpiw lrestvqgay gsedmkeggi 301 dmdafqeree ghagpddnee vmrallihek ktssamagsv gaaapvtaan gsdsesetse 361 sdddspprpa avavhkreed eeeddefeev addpivmvag rpfsysevsq rpelvaqmtp 421 eekeayiamg qrmfedlfe';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count36 = 0; count36 < 4500; count36++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdpsllrere lfkkralstp vvekrsasse sssssskkkk tkvehggssg skqnsdhsng 61 sfnlkalsgs sgykfgvlak ivnymktrhq rgdthpltld eildetqhld iglkqkqwlm 121 tealvnnpki evidgkyafk pkynvrdkka llrlldqhdq rglggilled ieealpnsqk 181 avkalgdqil fvnrpdkkki lffndkscqf svdeefqklw rsvtvdsmde ekieeylkrq 241 gissmqesgp kkvapiqrrk kpasqkkrrf kthnehlagv lkdysditss k';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

unused\_variable = 5;

if (unused\_variable == 5) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mklnvdgllv yfpydyiype qfsymrelkr tldakghgvl empsgtgktv sllalimayq 61 rayplevtkl iycsrtvpei ekvieelrkl lnfyekqege klpflglals srknlcihpe 121 vtplrfgkdv dgkchsltas yvraqyqhdt slphcrfyee fdahgrevpl pagiynlddl 181 kalgrrqgwc pyflarysil hanvvvysyh ylldpkiadl vskelarkav vvfdeahnid 241 nvcidsmsvn ltrrtldrcq gnletlqktv lriketdeqr lrdeyrrlve glreasaare 301 tdahlanpvl pdevlqeavp gsirtaehfl gflrrlleyv kwrlrvqhvv qesppaflsg 361 laqrvciqrk plrfcaerlr sllhtleitd ladfspltll anfatlvsty akgftiiiep 421 fddrtptian pilhfscmda slaikpvfer fqsviitsgt lspldiypki ldfhpvtmat 481 ftmtlarvcl cpmiigrgnd qvaisskfet rediavirny gnlllemsav vpdgivafft 541 syqymestva swyeqgilen iqrnkllfie tqdgaetsva lekyqeacen grgaillsva 601 rgkvsegidf vhhygravim fgvpyvytqs rilkarleyl rdqfqirend fltfdamrha 661 aqcvgrairg ktdyglmvfa dkrfargdkr gklprwiqeh ltdanlnltv degvqvakyf 721 lrqmaqpfhr edqlglslls leqleseetl krieqiaqql';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count38 = 0; count38 < 4500; count38++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkrdradrd kkksrkrhye deeddeedap gndpqeavps aagkqvdesg tkvdeygakd 61 yrlqmplkdd htsrplwvap dghifleafs pvykyaqdfl vaiaepvcrp thvheyklta 121 yslyaavsvg lqtsditeyl rklsktgvpd gimqfiklct vsygkvklvl khnryfvesc 181 hpdviqhllq dpvirecrlr nsegeateli tetftsksai sktaessggp stsrvtdpqg 241 ksdipmdlfd fyeqmdkdee eeeetqtvsf evkqemieel qkrcihleyp llaeydfrnd 301 svnpdinidl kptavlrpyq ekslrkmfgn grarsgvivl pcgagkslvg vtaactvrkr 361 clvlgnsavs veqwkaqfkm wstiddsqic rftsdakdkp igcsvaisty smlghttkrs 421 weaervmewl ktqewglmil devhtipakm frrvltivqa hcklgltatl vreddkivdl 481 nfligpklye anwmelqnng yiakvqcaev wcpmspefyr eyvaiktkkr illytmnpnk 541 fracqflikf herrndkiiv fadnvfalke yairlnkpyi ygptsqgerm qilqnfkhnp 601 kintifiskv gdtsfdlpea nvliqisshg gsrrqeaqrl grvlrakkgm vaeeynaffy 661 slvsqdtqem aystkrqrfl vdqgysfkvi tklagmeeed lafstkeeqq qllqkvlaat 721 dldaeeevva gefgsrssqa srrfgtmssm sgaddtvyme yhssrskaps khvhplfkrf 781 rk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matsseevll ivkkvrqkkq dgalylmaer iawapegkdr ftishmyadi kcqkispegk 61 akiqlqlvlh agdttnfhfs nestavkerd avkdllqqll pkfkrkanke leeknrmlqe 121 dpvlfqlykd lvvsqvisae efwanrlnvn atdssstsnh kqdvgisaaf ladvrpqtdg 181 cnglrynlts diiesifrty pavkmkyaen vphnmtekef wtrffqshyf hrdrlntgsk 241 dlfaecakid ekglktmvsl gvknplldlt aledkpldeg ygissvpsas nsksikensn 301 aaiikrfnhh samvlaaglr kqeaqneqts epsnmdgnsg dadcfqpavk raklqesiey 361 edlgknnsvk tialnlkksd ryyhgptpiq slqyatsqdi insfqsirqe meaytpkltq 421 vlsssaasst italspggal mqggtqqain qmvpndiqse lkhlyvavge llrhfwscfp 481 vntpfleekv vkmksnlerf qvtklcpfqe kirrqylstn lvshieemlq taynklhtwq 541 srrlmkkt';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count40 = 0; count40 < 4500; count40++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mestpsrgln rvhlqcrnlq eflgglspgv ldrlyghpat clavfrelps laknwvmrml 61 fleqplpqaa valwvkkefs kaqeestgll sglriwhtql lpgglqglil npifrqnlri 121 allgggkaws ddtsqlgpdk hardvpsldk yaeerwevvl hfmvgspsaa vsqdlaqlls 181 qaglmkstep geppcitsag fqfllldtpa qlwyfmlqyl qtaqsrgmdl veilsflfql 241 sfstlgkdys vegmsdslln flqhlrefgl vfqrkrksrr yyptrlainl ssgvsgaggt 301 vhqpgfivve tnyrlyayte selqialial fsemlyrfpn mvvaqvtres vqqaiasgit 361 aqqiihflrt rahpvmlkqt pvlpptitdq irlwelerdr lrftegvlyn qflsqvdfel 421 llaharelgv lvfensakrl mvvtpaghsd vkrfwkrqkh ss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdeepertkr weggyertwe ilkedesgsl katiedilfk akrkrvfehh gqvrlgmmrh 61 lyvvvdgsrt medqdlkpnr ltctlklley fveeyfdqnp isqigiivtk skraekltel 121 sgnprkhits lkkavdmtch gepslynsls iamqtlkhmp ghtsrevlii fsslttcdps 181 niydliktlk aakirvsvig lsaevrvctv laretggtyh vildeshyke llthhvsppp 241 assssecsli rmgfpqhtia slsdqdakps fsmahldgnt epgltlggyf cpqcrakyce 301 lpveckicgl tlvsaphlar syhhlfplda fqeipleeyn gerfcygcqg elkdqhvyvc 361 avcqnvfcvd cdvfvhdslh ccpgcihkip apsgv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count42 = 0; count42 < 4500; count42++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvsdedelnl lvivvdanpi wwgkqalkes qftlskcida vmvlgnshlf mnrsnklavi 61 ashiqesrfl ypgkngrlgd ffgdpgnppe fnpsgskdgk yelltsanev iveeikdlmt 121 ksdikgqhte tllagslaka lcyihrmnke vkdnqemksr ilvikaaeds alqymnfmnv 181 ifaaqkqnil idacvldsds gllqqacdit gglylkvpqm psllqyllwv flpdqdqrsq 241 lilpppvhvd yraacfchrn lieigyvcsv clsifcnfsp icttcetafk islppvlkak 301 kkklkvsa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvnvlkgvli ecdpamkqfl lyldesnalg kkfiiqdidd thvfviaelv nvlqervgel 61 mdqnafsltq k';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count44 = 0; count44 < 4500; count44++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maldvksrak ryekldflge gqfatvykar dkntnqivai kkiklghrse akdginrtal 61 reikllqels hpniigllda fghksnislv fdfmetdlev iikdnslvlt pshikaymlm 121 tlqgleylhq hwilhrdlkp nnllldengv lkladfglak sfgspnrayt hqvvtrwyra 181 pellfgarmy gvgvdmwavg cilaelllrv pflpgdsdld qltrifetlg tpteeqwpdm 241 cslpdyvtfk sfpgiplhhi fsaagddlld liqglflfnp caritatqal kmkyfsnrpg 301 ptpgcqlprp ncpvetlkeq snpalaikrk rtealeqggl pkklif';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mddqgcprck ttkyrnpslk lmvnvcghtl cescvdllfv rgagncpecg tplrksnfrv 61 qlfedptvdk eveirkkvlk iynkreedfp slreyndfle eveeivfnlt nnvdldntkk 121 kmeiyqkenk dviqknklkl treqeeleea leverqeneq rrlfiqkeeq lqqilkrknk 181 qafldeless dlpvalllaq hkdrstqlem qlekpkpvkp vtfstgikmg qhislapihk 241 leealyeyqp lqietygphv pelemlgrlg ylnhvraasp qdlaggytss lachralqda 301 fsglfwqps';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count46 = 0; count46 < 4500; count46++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myhnssqkrh wtfsseeqla rlradanrkf rckavangkv lpndpvflep heemtlckyy 61 ekrllefcsv fkpamprsvv gtacmyfkrf ylnnsvmeyh priimltcaf lackvdefnv 121 sspqfvgnlr esplgqekal eqileyelll iqqlnfhliv hnpyrpfegf lidlktrypi 181 lenpeilrkt addflnrial tdayllytps qialtailss asragitmes ylseslmlke 241 nrtclsqlld imksmrnlvk kyepprseev avlkqklerc hsaelalnvi tkkrkgyedd 301 dyvskkskhe eeewtdddlv esl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

unused\_variable = 5;

if (unused\_variable == 5) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 makqydsvec pfcdevskye klakigqgtf gevfkarhrk tgqkvalkkv lmenekegfp 61 italreikil qllkhenvvn lieicrtkas pynrckgsiy lvfdfcehdl agllsnvlvk 121 ftlseikrvm qmllnglyyi hrnkilhrdm kaanvlitrd gvlkladfgl arafslakns 181 qpnrytnrvv tlwyrppell lgerdygppi dlwgagcima emwtrspimq gnteqhqlal 241 isqlcgsitp evwpnvdnye lyeklelvkg qkrkvkdrlk ayvrdpyald lidkllvldp 301 aqridsddal nhdffwsdpm psdlkgmlst hltsmfeyla pprrkgsqit qqstnqsrnp 361 attnqtefer vf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count48 = 0; count48 < 4500; count48++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 megerknnnk rwyftreqle nspsrrfgvd pdkelsyrqq aanllqdmgq rlnvsqltin 61 taivymhrfy miqsftqfpg nsvapaalfl aakveeqpkk lehvikvaht clhpqeslpd 121 trseaylqqv qdlvilesii lqtlgfelti dhphthvvkc tqlvraskdl aqtsyfmatn 181 slhlttfslq ytppvvacvc ihlackwsnw eipvstdgkh wweyvdatvt lelldelthe 241 flqilektpn rlkriwnwra ceaakktkad drgtdektse qtilnmisqs ssdttiaglm 301 smststtsav pslpvseess snltsvemlp gkrwlssqps fkleptqghr tsenlaltgv 361 dhslpqdgsn afisqkqnsk svpsakvslk eyrakhaeel aaqkrqlenm eanvksqyay 421 aaqnllshhd shssvilkmp iegsenperp flekadktal kmripvaggd kaasskpeei 481 kmrikvhaaa dkhnsvedsv tksrehkekh kthpsnhhhh hnhhshkhsh sqlpvgtgnk 541 rpgdpkhssq tsnlahktys lsssfsssss trkrgpseet ggavfdhpak iakstksssl 601 nfsfpslptm gqmpghssdt sglsfsqpsc ktrvphskld kgptganghn ttqtidyqdt 661 vnmlhsllsa qgvqptqpta fefvrpysdy lnprsggiss rsgntdkprp pplpsepppp 721 lpplpk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masgrgassr wfftreqlen tpsrrcgvea dkelscrqqa anliqemgqr lnvsqltint 61 aivymhrfym hhsftkfnkn iisstalfla akveeqarkl ehvikvahac lhpleplldt 121 kcdaylqqtq elviletiml qtlgfeitie hphtdvvkct qlvraskdla qtsyfmatns 181 lhlttfclqy kptviacvci hlackwsnwe ipvstdgkhw weyvdptvtl elldelthef 241 lqilektpnr lkkirnwran qaarkpkvdg qvsetpllgs slvqnsilvd svtgvptnps 301 fqkpstsafp apvplnsgni svqdshtsdn lsmlatgmps tsyglsshqe wpqhqdsart 361 eqlysqkqet slsgsqynin fqqgpsislh sglhhrpdki sdhssvkqey thkagsskhh 421 gpisttpgii pqkmsldkyr ekrkletldl dvrdhyiaaq veqqhkqgqs qaassssvts 481 pikmkipian tekymadkke ksgslklrip ipptdksask eelkmkikvs sserhsssde 541 gsgkskhssp hisrdhkekh kehpssrhht sshkhshshs gsssggskhs adgipptvlr 601 spvglssdgi sssssssrkr lhvndashnh hskmskssks sgglrtsqhp retgqeasgd 661 qrs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count50 = 0; count50 < 4500; count50++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkenkenssp svtsanldht kpcwywdkkd lahtpsqleg ldpatearyr regarfifdv 61 gtrlglhydt latgiiyfhr fymfhsfkqf pryvtgaccl flagkveetp kkckdiikta 121 rsllndvqfg qfgddpkeev mvlerillqt ikfdlqvehp yqfllkyakq lkgdknkiqk 181 lvqmawtfvn dslcttlslq wepeiiavav mylagrlckf eiqewtskpm yrrwweqfvq 241 dvpvdvledi chqildlysq gkqqmphhtp hqlqqppslq ptpqvpqvqq sqpsqsseps 301 qpqqkdpqqp aqqqqpaqqp kkpspqpssp rqvkravvvs pkeenkaaep pppkipkiet 361 thpplppahp ppdrkpplaa algeaeppgp vdatdlpkvq ipppahpapv hqppplphrp 421 pppppssymt gmsttssyms gegyqslqsm mktegpsyga lppaygppah lpyhphvypp 481 npppppvppp pasfpppaip pptpgypppp ptynpnfppp pprlppthav pphpppglgl 541 ppasypppav ppggqppvpp pipppgmppv gglgraawmr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 medevvrfak kmdkmvqkkn aagaldllke lknipmtlel lqstrigmsv nairkqstde 61 evtslaksli kswkklldgp stekdldekk kepaitsqns peareestss gnvsnrkdet 121 nardtyvssf prapstsdsv rlkcremlaa alrtgddyia igadeeelgs qieeaiyqei 181 rntdmkyknr vrsrisnlkd aknpnlrknv lcgnippdlf armtaeemas delkemrknl 241 tkeairehqm aktggtqtdl ftcgkckkkn ctytqvqtrs adepmttfvv cnecgnrwkf 301 c';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count52 = 0; count52 < 4500; count52++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmgkeeeiar iarrldkmvt kksaegamdl lrelkampit lhllqstrvg msvnalrkqs 61 sdeevialak slikswkkll dasdakarer grgmplptss rdaseapdps rkrpelprap 121 stprittfpp vpvtcdavrn kcremltaal qtdhdhvaig adcerlsaqi eecifrdvgn 181 tdmkyknrvr srisnlkdak npdlrrnvlc gaitpqqiav mtseemasde lkeirkamtk 241 eairehqmar tggtqtdlft cgkcrkknct ytqvqtrssd epmttfvvcn ecgnrwkfc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myavykqahp ptglefsmyc nffnnsernl vvagtsqlyv yrlnrdaeal tkndrstegk 61 ahreklelaa sfsffgnvms masvqlagak rdalllsfkd aklsvveydp gthdlktlsl 121 hyfeepelrd gfvqnvhtpr vrvdpdgrca amlvygtrlv vlpfrresla eeheglvgeg 181 qrssflpsyi idvraldekl lniidlqflh gyyeptllil fepnqtwpgr vavrqdtcsi 241 vaislnitqk vhpviwslts lpfdctqala vpkpiggvvv favnsllyln qsvppygval 301 nslttgttaf plrtqegvri tldcaqatfi sydkmvislk ggeiyvltli tdgmrsvraf 361 hfdkaaasvl ttsmvtmepg ylflgsrlgn slllkytekl qeppasavre aadkeeppsk 421 kkrvdatagw saagksvpqd evdeievygs eaqsgtqlat ysfevcdsil nigpcanaav 481 gepaflseef qnspepdlei vvcsghgkng alsvlqksir pqvvttfelp gcydmwtvia 541 pvrkeeednp kgegteqeps ttpeadddgr rhgflilsre dstmilqtgq eimeldtsgf 601 atqgptvfag nigdnryivq vsplgirlle gvnqlhfipv dlgapivqca vadpyvvims 661 aeghvtmfll ksdsyggrhh rlalhkpplh hqskvitlcl yrdlsgmftt esrlggarde 721 lggrsgpeae glgsetsptv ddeeemlygd sgslfspske earrssqppa drdpapfrae 781 pthwcllvre ngtmeiyqlp dwrlvflvkn fpvgqrvlvd ssfgqpttqg earreeatrq 841 gelplvkevl lvalgsrqsr pyllvhvdqe lliyeafphd sqlgqgnlkv rfkkvphnin 901 frekkpkpsk kkaegggaee gagargrvar fryfediygy sgvficgpsp hwllvtgrga 961 lrlhpmaidg pvdsfapfhn vncprgflyf nrqgelrisv lpaylsydap wpvrkiplrc 1021 tahyvayhve skvyavatst ntpcariprm tgeekefeti erderyihpq qeafsiqlis 1081 pvsweaipna rielqewehv tcmktvslrs eetvsglkgy vaagtclmqg eevtcrgril 1141 imdvievvpe pgqpltknkf kvlyekeqkg pvtalchcng hlvsaigqki flwslrasel 1201 tgmafidtql yihqmisvkn filaadvmks isllryqees ktlslvsrda kplevysvdf 1261 mvdnaqlgfl vsdrdrnlmv ymylpeakes fggmrllrra dfhvgahvnt fwrtpcrgat 1321 eglskksvvw enkhitwfat ldggiglllp mqektyrrll mlqnalttml phhaglnpra 1381 frmlhvdrrt lqnavrnvld gellnrylyl stmerselak kigttpdiil ddlletdrvt 1441 ahf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count54 = 0; count54 < 4500; count54++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtsiiklttl sgvqeesalc yllqvdefrf lldcgwdehf smdiidslrk hvhqidavll 61 shpdplhlga lpyavgklgl ncaiyatipv ykmgqmfmyd lyqsrhnted ftlftlddvd 121 aafdkiqqlk fsqivnlkgk ghglsitplp aghmiggtiw kivkdgeeei vyavdfnhkr 181 eihlngcsle mlsrpsllit dsfnatyvqp rrkqrdeqll tnvletlrgd gnvliavdta 241 grvlelaqll dqiwrtkdag lgvyslalln nvsynvvefs ksqvewmsdk lmrcfedkrn 301 npfqfrhlsl chglsdlarv pspkvvlasq pdlecgfsrd lfiqwcqdpk nsiiltyrtt 361 pgtlarflid npsekiteie lrkrvklegk eleeylekek lkkeaakkle qskeadidss 421 desdieedid qpsahktkhd lmmkgegsrk gsffkqakks ypmfpapeer ikwdeygeii 481 kpedflvpel qateeekskl esgltngdep mdqdlsdvpt kcisttesie ikarvtyidy 541 egrsdgdsik kiinqmkprq liivhgppea sqdlaeccra fggkdikvym pklhetvdat 601 sethiyqvrl kdslvsslqf ckakdaelaw idgvldmrvs kvdtgvilee gelkddgeds 661 emqveapsds sviaqqkamk slfgddeket geeseiiptl eplpphevpg hqsvfmnepr 721 lsdfkqvllr egiqaefvgg vlvcnnqvav rrtetgrigl egclcqdfyr irdllyeqya 781 iv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msaipaeesd qllirplgag qevgrsciil efkgrkimld cgihpglegm dalpyidlid 61 paeidlllis hfhldhcgal pwflqktsfk grtfmthatk aiyrwllsdy vkvsnisadd 121 mlytetdlee smdkietinf hevkevagik fwcyhaghvl gaamfmieia gvkllytgdf 181 srqedrhlma aeipnikpdi liiestygth ihekreerea rfcntvhdiv nrggrglipv 241 falgraqell lildeywqnh pelhdipiyy asslakkcma vyqtyvnamn dkirkqinin 301 npfvfkhisn lksmdhfddi gpsvvmaspg mmqsglsrel feswctdkrn gviiagycve 361 gtlakhimse peeittmsgq klplkmsvdy isfsahtdyq qtsefiralk pphvilvhge 421 qnemarlkaa lireyednde vhievhnprn teavtlnfrg eklakvmgfl adkkpeqgqr 481 vsgilvkrnf nyhilspcdl snytdlamst vkqtqaipyt gpfnllcyql qkltgdveel 541 eiqekpalkv fknitviqep gmvvlewlan psndmyadtv ttvilevqsn pkirkgavqk 601 vskklemhvy skrleimlqd ifgedcvsvk ddsilsvtvd gktanlnlet rtveceegse 661 ddeslremve laaqrlyeal tpvh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count56 = 0; count56 < 4500; count56++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqeiiasvdh ikfdleiave qqlgaqplpf pgmdksgaav cefflkaacg kggmcpfrhi 61 sgektvvckh wlrglckkgd qceflheydm tkmpecyfys kfgecsnkec pflhidpesk 121 ikdcpwydrg fckhgplcrh rhtrrvicvn ylvgfcpegp sckfmhprfe lpmgtteqpp 181 lpqqtqppak qsnnpplqrs ssliqltsqn sspnqqrtpq vigvmqsqns sagnrgprpl 241 eqvtcykcge kghyanrctk ghlaflsgq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msvvppnrsq tgwprgvtqf gnkyiqqtkp ltlertinly pltnytfgtk eplyekdssv 61 aarfqrmree fdkigmrrtv egvlivhehr lphvlllqlg ttffklpgge lnpgedeveg 121 lkrlmteilg rqdgvlqdwv iddcignwwr pnfeppqypy ipahitkpke hkklflvqlq 181 ekalfavpkn yklvaaplfe lydnapgygp iisslpqlls rfnfiyn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count58 = 0; count58 < 4500; count58++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madgvdhidi yadvgeefnq eaeygghdqi dlyddvisps anngdapedr dymdtlpptv 61 gddvgkgaap nvvytytgkr ialyignltw wttdedltea vhslgvndil eikffenran 121 gqskgfalvg vgseasskkl mdllpkrelh gqnpvvtpcn kqflsqfemq srkttqsgqm 181 sgegkagppg gssraafpqg grgrgrfpga vpggdrfpgp agpggppppf pagqtpprpp 241 lgppgppgpp gppppgqvlp pplagppnrg drppppvlfp gqpfgqpplg plppgppppv 301 pgygpppgpp ppqqgppppp gpfpprppgp lgppltlapp phlpgpppga pppaphvnpa 361 ffppptnsgm ptsdsrgppp tdpygrpppy drgdygppgr emdtartpls eaefeeimnr 421 nraisssais ravsdasagd ygsaietlvt aislikqskv saddrckvli sslqdclhgi 481 esksygsgsr rersrerdhs rsreksrrhk srsrdrhddy yrersrerer hrdrdrdrdr 541 erdrereyrh r';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mansakaeey ekmsleqaka svnsetessf ninenttasg tglsektsvc rqvdiarkrk 61 efeddlvkes sscgkdtpsk krkldpeivp eekdcgdaeg nskkrkrete dvpkdksstg 121 dgtqnkrkia ledvpekqkn leeghsstva ahynelqevg lekrsqsrif ylrnfnnwmk 181 svligeflek vrqkkkrdit vldlgcgkgg dllkwkkgri nklvctdiad vsvkqcqqry 241 edmknrrdse yifsaefita dsskellidk frdpqmcfdi cscqfvchys fesyeqadmm 301 lrnacerlsp ggyfigttpn sfelirrlea setesfgnei ytvkfqkkgd yplfgckydf 361 nlegvvdvpe flvyfpllne makkynmklv ykktflefye ekiknnenkm llkrmqalep 421 ypanessklv sekvddyeha akymknsqvr lplgtlskse weatrltvti mreawlstvg 481 pgrapvaass vkwgtprpam qfil';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count60 = 0; count60 < 4500; count60++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mahnkipprw lncprrgqpv agrflplktm lgprydsqva eenrfhpsml snylkslkvk 61 mgllvdltnt srfydrndie kegikyiklq ckghgecptt entetfirlc erfnernppe 121 ligvhcthgf nrtgflicaf lvekmdwsie aavatfaqar ppgiykgdyl kelfrrygdi 181 eeappppllp dwcfeddede dededgkkes epgssasfgk rrkerlklga iflegvtvkg 241 vtqvttqpkl gevqqkchqf cgwegsgfpg aqpvsmdkqn iklldlkpyk vswkadgtry 301 mmlidgtnev fmidrdnsvf hvsnlefpfr kdlrmhlsnt lldgemiidr vngqavpryl 361 iydiikfnsq pvgdcdfnvr lqciereiis prhekmktgl idktqepfsv rnkpffdict 421 srkllegnfa kevshemdgl ifqptgkykp grcddilkwk ppslnsvdfr lkitrmggeg 481 llpqnvglly vggyerpfaq ikvtkelkqy dnkiieckfe nnswvfmrqr tdksfpnayn 541 tamavcnsis npvtkemlfe fidrctaasq gqkrkhhldp dtelmppppp krprplt';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mansakaeey ekmsleqaka svnsetessf ninenttasg tglsektsvc rqvdiarkrk 61 efeddlvkes sscgkdtpsk krkldpeivp eekdcgdaeg nskkrkrete dvpkdksstg 121 dgtqnkrkia ledvpekqkn leeghsstva ahynelqevg lekrsqsrif ylrnfnnwmk 181 svligeflek vrqkkkrdit vldlgcgkgg dllkwkkgri nklvctdiad vsvkqcqqry 241 edmknrrdse yifsaefita dsskellidk frdpqmcfdi cscqfvchys fesyeqadmm 301 lrnacerlsp ggyfigttpn sfelirrlea setesfgnei ytvkfqkkgd yplfgckydf 361 nlegvvdvpe flvyfpllne makkynmklv ykktflefye ekiknnenkm llkrmqalep 421 ypanessklv sekvddyeha akymknsqvr lplgtlskse weatrltvti mreawlstvg 481 pgrapvaass vkwgtprpam qfil';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count62 = 0; count62 < 4500; count62++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myavykqahp ptglefsmyc nffnnsernl vvagtsqlyv yrlnrdaeal tkndrstegk 61 ahreklelaa sfsffgnvms masvqlagak rdalllsfkd aklsvveydp gthdlktlsl 121 hyfeepelrd gfvqnvhtpr vrvdpdgrca amlvygtrlv vlpfrresla eeheglvgeg 181 qrssflpsyi idvraldekl lniidlqflh gyyeptllil fepnqtwpgr vavrqdtcsi 241 vaislnitqk vhpviwslts lpfdctqala vpkpiggvvv favnsllyln qsvppygval 301 nslttgttaf plrtqegvri tldcaqatfi sydkmvislk ggeiyvltli tdgmrsvraf 361 hfdkaaasvl ttsmvtmepg ylflgsrlgn slllkytekl qeppasavre aadkeeppsk 421 kkrvdatagw saagksvpqd evdeievygs eaqsgtqlat ysfevcdsil nigpcanaav 481 gepaflseef qnspepdlei vvcsghgkng alsvlqksir pqvvttfelp gcydmwtvia 541 pvrkeeednp kgegteqeps ttpeadddgr rhgflilsre dstmilqtgq eimeldtsgf 601 atqgptvfag nigdnryivq vsplgirlle gvnqlhfipv dlgapivqca vadpyvvims 661 aeghvtmfll ksdsyggrhh rlalhkpplh hqskvitlcl yrdlsgmftt esrlggarde 721 lggrsgpeae glgsetsptv ddeeemlygd sgslfspske earrssqppa drdpapfrae 781 pthwcllvre ngtmeiyqlp dwrlvflvkn fpvgqrvlvd ssfgqpttqg earreeatrq 841 gelplvkevl lvalgsrqsr pyllvhvdqe lliyeafphd sqlgqgnlkv rfkkvphnin 901 frekkpkpsk kkaegggaee gagargrvar fryfediygy sgvficgpsp hwllvtgrga 961 lrlhpmaidg pvdsfapfhn vncprgflyf nrqgelrisv lpaylsydap wpvrkiplrc 1021 tahyvayhve skvyavatst ntpcariprm tgeekefeti erderyihpq qeafsiqlis 1081 pvsweaipna rielqewehv tcmktvslrs eetvsglkgy vaagtclmqg eevtcrgril 1141 imdvievvpe pgqpltknkf kvlyekeqkg pvtalchcng hlvsaigqki flwslrasel 1201 tgmafidtql yihqmisvkn filaadvmks isllryqees ktlslvsrda kplevysvdf 1261 mvdnaqlgfl vsdrdrnlmv ymylpeakes fggmrllrra dfhvgahvnt fwrtpcrgat 1321 eglskksvvw enkhitwfat ldggiglllp mqektyrrll mlqnalttml phhaglnpra 1381 frmlhvdrrt lqnavrnvld gellnrylyl stmerselak kigttpdiil ddlletdrvt 1441 ahf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtsiiklttl sgvqeesalc yllqvdefrf lldcgwdehf smdiidslrk hvhqidavll 61 shpdplhlga lpyavgklgl ncaiyatipv ykmgqmfmyd lyqsrhnted ftlftlddvd 121 aafdkiqqlk fsqivnlkgk ghglsitplp aghmiggtiw kivkdgeeei vyavdfnhkr 181 eihlngcsle mlsrpsllit dsfnatyvqp rrkqrdeqll tnvletlrgd gnvliavdta 241 grvlelaqll dqiwrtkdag lgvyslalln nvsynvvefs ksqvewmsdk lmrcfedkrn 301 npfqfrhlsl chglsdlarv pspkvvlasq pdlecgfsrd lfiqwcqdpk nsiiltyrtt 361 pgtlarflid npsekiteie lrkrvklegk eleeylekek lkkeaakkle qskeadidss 421 desdieedid qpsahktkhd lmmkgegsrk gsffkqakks ypmfpapeer ikwdeygeii 481 kpedflvpel qateeekskl esgltngdep mdqdlsdvpt kcisttesie ikarvtyidy 541 egrsdgdsik kiinqmkprq liivhgppea sqdlaeccra fggkdikvym pklhetvdat 601 sethiyqvrl kdslvsslqf ckakdaelaw idgvldmrvs kvdtgvilee gelkddgeds 661 emqveapsds sviaqqkamk slfgddeket geeseiiptl eplpphevpg hqsvfmnepr 721 lsdfkqvllr egiqaefvgg vlvcnnqvav rrtetgrigl egclcqdfyr irdllyeqya 781 iv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count64 = 0; count64 < 4500; count64++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msaipaeesd qllirplgag qevgrsciil efkgrkimld cgihpglegm dalpyidlid 61 paeidlllis hfhldhcgal pwflqktsfk grtfmthatk aiyrwllsdy vkvsnisadd 121 mlytetdlee smdkietinf hevkevagik fwcyhaghvl gaamfmieia gvkllytgdf 181 srqedrhlma aeipnikpdi liiestygth ihekreerea rfcntvhdiv nrggrglipv 241 falgraqell lildeywqnh pelhdipiyy asslakkcma vyqtyvnamn dkirkqinin 301 npfvfkhisn lksmdhfddi gpsvvmaspg mmqsglsrel feswctdkrn gviiagycve 361 gtlakhimse peeittmsgq klplkmsvdy isfsahtdyq qtsefiralk pphvilvhge 421 qnemarlkaa lireyednde vhievhnprn teavtlnfrg eklakvmgfl adkkpeqgqr 481 vsgilvkrnf nyhilspcdl snytdlamst vkqtqaipyt gpfnllcyql qkltgdveel 541 eiqekpalkv fknitviqep gmvvlewlan psndmyadtv ttvilevqsn pkirkgavqk 601 vskklemhvy skrleimlqd ifgedcvsvk ddsilsvtvd gktanlnlet rtveceegse 661 ddeslremve laaqrlyeal tpvh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqeiiasvdh ikfdleiave qqlgaqplpf pgmdksgaav cefflkaacg kggmcpfrhi 61 sgektvvckh wlrglckkgd qceflheydm tkmpecyfys kfgecsnkec pflhidpesk 121 ikdcpwydrg fckhgplcrh rhtrrvicvn ylvgfcpegp sckfmhprfe lpmgtteqpp 181 lpqqtqppak qsnnpplqrs ssliqltsqn sspnqqrtpq vigvmqsqns sagnrgprpl 241 eqvtcykcge kghyanrctk ghlaflsgq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count66 = 0; count66 < 4500; count66++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madgvdhidi yadvgeefnq eaeygghdqi dlyddvisps anngdapedr dymdtlpptv 61 gddvgkgaap nvvytytgkr ialyignltw wttdedltea vhslgvndil eikffenran 121 gqskgfalvg vgseasskkl mdllpkrelh gqnpvvtpcn kqflsqfemq srkttqsgqm 181 sgegkagppg gssraafpqg grgrgrfpga vpggdrfpgp agpggppppf pagqtpprpp 241 lgppgppgpp gppppgqvlp pplagppnrg drppppvlfp gqpfgqpplg plppgppppv 301 pgygpppgpp ppqqgppppp gpfpprppgp lgppltlapp phlpgpppga pppaphvnpa 361 ffppptnsgm ptsdsrgppp tdpygrpppy drgdygppgr emdtartpls eaefeeimnr 421 nraisssais ravsdasagd ygsaietlvt aislikqskv saddrckvli sslqdclhgi 481 esksygsgsr rersrerdhs rsreksrrhk srsrdrhddy yrersrerer hrdrdrdrdr 541 erdrereyrh r';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgrpesaggg srgpfegggr arraggiflt lsilrtrdlp sgamsegvdl idiyadeefn 61 qdpefnntdq idlyddvlta tsqpsddrss steppppvrq epspkpnnkt pailytysgl 121 rnrraavyvg sfswwttdqq liqvirsigv ydvvelkfae nrangqskgy aevvvasens 181 vhkllellpg kvlngekvdv rpatrqnlsq feaqarkrec vrvprggipp rahsrdssds 241 adgratpsen lvpssarvdk ppsvlpyfnr ppsalplmgl ppppippppp lsssfgvppp 301 ppgihyqhlm pppprlpphl avpppgaipp alhlnpaffp ppnatvgppp dtymkasapy 361 nhhgsrdsgp ppstvseaef edimkrnrai sssaiskavs gasagdysda ietlltaiav 421 ikqsrvande rcrvlisslk dclhgieaks ysvgasgsss rkrhrsrers psrsressrr 481 hrdllhnedr hddyfqernr eherhrdrer drhh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count68 = 0; count68 < 4500; count68++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqeviagler ftfafekdve mqkgtgllpf qgmdksasav cnfftkglce kgklcpfrhd 61 rgekmvvckh wlrglckkgd hckflhqydl trmpecyfys kfgdcsnkec sflhvkpafk 121 sqdcpwydqg fckdgplcky rhvprimcln ylvgfcpegp kcqfaqkire fkllpgski';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 myrtkvglkd rqqlykliis qllydgyisi anglineikp qsvcapseql lhliklgmen 61 ddtavqyaig rsdtvapgtg idlefdadvq tmspeaseye tcyvtshkgp crvatysrdg 121 qliatgsada sikildterm laksampiev mmnetaqqnm enhpvirtly dhvdevtcla 181 fhpteqilas gsrdytlklf dyskpsakra fkyiqeaeml rsisfhpsgd filvgtqhpt 241 lrlydintfq cfvscnpqdq htdaicsvny nssanmyvtg skdgciklwd gvsnrcittf 301 ekahdgaevc saifsknsky ilssgkdsva klweistgrt lvrytgagls grqvhrtqav 361 fnhtedyvll pdertislcc wdsrtaerrn llslghnniv rcivhsptnp gfmtcsddfr 421 arfwyrrstt d';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count70 = 0; count70 < 4500; count70++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 magltvrdpa vdrslrsvfv gnipyeatee qlkdifsevg pvvsfrlvyd retgkpkgyg 61 fceyqdqeta lsamrnlngr efsgralrvd naaseknkee lkslgtgapv iespygetis 121 pedapesisk avaslppeqm felmkqmklc vqnspqearn mllqnpqlay allqaqvvmr 181 ivdpeialki lhrqtniptl iagnpqpvhg agpgsgsnvs mnqqnpqapq aqslggmhvn 241 gapplmqasm qggvpapgqm paavtgpgpg slapgggmqa qvgmpgsgpv smergqgtlq 301 hspvgpagpa siervqvpmq dpraamqrgs lpanvptprg llgdapndpr ggtllsvtge 361 veprgylgpp hqgppmhhvp ghesrgppph elrggplpep rplmaeprgp mldqrgppld 421 grggrdprgi dargmearam eargldargl earamearam earamearam earamevrgm 481 eargmdtrgp vpgprgpips gmqgpspinm gavvpqgsrq vpvmqgtgmq gasiqggsqp 541 ggfspgqnqv tpqdhekaal imqvlqltad qiamlppeqr qsililkeqi qkstgap';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msgdgateqa aeyvpekvkk aekkleenpy dldawsilir eaqnqpidka rktyerlvaq 61 fpssgrfwkl yieaeikakn ydkveklfqr clmkvlhidl wkcylsyvre tkgklpsyke 121 kmaqaydfal dkigmeimsy qiwvdyinfl kgveavgsya enqritavrr vyqrgcvnpm 181 inieqlwrdy nkyeeginih lakkmiedrs rdymnarrva keyetvmkgl drnapsvppq 241 ntpqeaqqvd mwkkyiqwek snplrtedqt litkrvmfay eqcllvlghh pdiwyeaaqy 301 leqsskllae kgdmnnaklf sdeaaniyer aistllkknm llyfayadye esrmkyekvh 361 siynrllaie didptlvyiq ymkfarraeg iksgrmifkk aredtrtrhh vyvtaalmey 421 ycskdksvaf kifelglkky gdipeyvlay idylshlned nntrvlferv ltsgslppek 481 sgeiwarfla fesnigdlas ilkvekrrft afkeeyegke tallvdrykf mdlypcsase 541 lkalgykdvs raklaaiipd pvvapsivpv lkdevdrkpe ypkpdtqqmi pfqprhlapp 601 glhpvpggvf pvppaavvlm kllpppicfq gpfvqvdelm eifrrckipn tveeavriit 661 ggapelaveg ngpvesnavl tkavkrpned sdedeekgav vppvhdiyra rqqkrir';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count72 = 0; count72 < 4500; count72++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msvvppnrsq tgwprgvtqf gnkyiqqtkp ltlertinly pltnytfgtk eplyekdssv 61 aarfqrmree fdkigmrrtv egvlivhehr lphvlllqlg ttffklpgge lnpgedeveg 121 lkrlmteilg rqdgvlqdwv iddcignwwr pnfeppqypy ipahitkpke hkklflvqlq 181 ekalfavpkn yklvaaplfe lydnapgygp iisslpqlls rfnfiyn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpfpvttqgs qqtqppqkhy gitspislaa pketdcvltq klietlkpfg vfeeeeelqr 61 rililgklnn lvkewireis esknlpqsvi envggkiftf gsyrlgvhtk gadidalcva 121 prhvdrsdff tsfydklklq eevkdlrave eafvpviklc fdgieidilf arlalqtipe 181 dldlrddsll knldircirs lngcrvtdei lhlvpnidnf rltlraiklw akrhniysni 241 lgflggvswa mlvartcqly pnaiastlvh kfflvfskwe wpnpvllkqp eecnlnlpvw 301 dprvnpsdry hlmpiitpay pqqnstynvs vstrmvmvee fkqglaitde illskaewsk 361 lfeapnffqk ykhyivllas aptekqrlew vglveskiri lvgsleknef itlahvnpqs 421 fpapkenpdk eefrtmwvig lvfkktense nlsvdltydi qsftdtvyrq ainskmfevd 481 mkiaamhvkr kqlhqllpnh vlqkkkkhst egvkltalnd ssldlsmdsd nsmsvpspts 541 atktsplnss gssqgrnspa pavtaasvtn iqatevsvpq vnssessggt ssesipqtat 601 qpaispppkp tvsrvvsstr lvnppprssg naatsgnaat kiptpivgvk rtssphkees 661 pkktkteede tsedanclal sghdkteake qldtetsttq setiqtaasl lasqktsstd 721 lsdipalpan pipviknsik lrlnr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count74 = 0; count74 < 4500; count74++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnpsapsypm aslyvgdlhp dvteamlyek fspagpilsi rvcrdmitrr slgyayvnfq 61 qpadaerald tmnfdvikgk pvrimwsqrd pslrksgvgn ifiknldksi dnkalydtfs 121 afgnilsckv vcdengskgy gfvhfetqea aeraiekmng mllndrkvfv grfksrkere 181 aelgarakef tnvyiknfge dmdderlkdl fgkfgpalsv kvmtdesgks kgfgfvsfer 241 hedaqkavde mngkelngkq iyvgraqkkv erqtelkrkf eqmkqdritr yqgvnlyvkn 301 lddgidderl rkefspfgti tsakvmmegg rskgfgfvcf sspeeatkav temngrivat 361 kplyvalaqr keerqahltn qymqrmasvr avpnpvinpy qpappsgyfm aaipqtqnra 421 ayyppsqiaq lrpsprwtaq garphpfqnm pgairpaapr ppfstmrpas sqvprvmstq 481 rvantstqtm gprpaaaaaa atpavrtvpq ykyaagvrnp qqhlnaqpqv tmqqpavhvq 541 gqepltasml asappqeqkq mlgerlfpli qamhptlagk itgmlleidn sellhmlesp 601 eslrskvdea vavlqahqak eaaqkavnsa tgvptv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mseqtpaeag aagaredacr dyqssledlt fnskphinml tilaeenlpf akeivsliea 61 qtakapssek lpvmylmdsi vknvgreylt aftknlvatf icvfekvden trkslfklrs 121 twdeifplkk lyaldvrvns ldpawpikpl ppnvntssih vnpkflnksp eepstpgtvv 181 sspsistppi vpdiqknltq eqlirqqlla kqkqllelqq kkleleleqa kaqlavslsv 241 qqetsnlgpg sapsklhvsq ippmavkaph qvpvqseksr pgpslqiqdl kgtnrdprln 301 risqhshgkd qshrkeflmn tlnqsdtkts ktipseklns skqeksksge kitkkeldql 361 dsksksksks psplknklsh tkdlknqese smrlsdmnkr dprlkkhlqd ktdgkdddvk 421 ekrktaekkd kdehmksseh rlagsrnkii ngivqkqdti teesekqgtk pgrsstrkrs 481 rsrspksrsp iihspkrrdr rspkrrqrsm sptstpkagk irqsgakqsh meeftppsre 541 drnakrstkq dirdprrmkk teeerpqett nqhstksgte pkenvenwqs sksakrwksg 601 weenkslqqv dehskpphlr hreswsstkg ilsprapkqq qhrlsvdanl qipkeltlas 661 krellqktse rlasgeitqd dflvvvhqir qlfqyqegkh rcnvrdspte enkgglkkkp 721 llsdaeltyy ehkaklkrtq vqhsfprldl ldpdifdypl tdallsgiec epskskhasr 781 nsgaqfdrke qfserarrls pisgsrtyae nlsphegrrr hdeqvsakgv reeqrspfnd 841 rfplkrprye dsdkpfvdsp asrfagldtn qrltalaedr plfdgpsrps vardgptkmi 901 fegpnklspr idgpptpasl rfdgspgqmg gggplrfegp qgqlgggcpl rfegppgpvg 961 tplrfegpig qaggggfrfe gspglrfegs pgglrfegpg gqpvgglrfe ghrgqpvggl 1021 rfegphgqpv gglrfdnprg qpvgglrfeg ghgpsgaair fdgphgqpgg girfegpllq 1081 qgvgmrfegp hgqsvaglrf egqhnqlggn lrfegphgqp gvgirfegpl vqqgggmrfe 1141 gpsvpggglr iegplgqggp rfegchalrf dgqpgqpsll prfdglhgqp gprfertpgq 1201 pgpqrfdgpp gqqvqprfdg vpqrfdgpqh qqasrfdipl glqgtrfdnh psqrlesvsf 1261 nqtgpyndpp gnafnapsqg lqfqrheqif dspqgpnfng phgpgnqsfs nplnrasghy 1321 fdeknlqssq fgnfgnipap mtvgniqasq qvlsgvaqpv afgqgqqflp vhpqnpgfvq 1381 npsgalpkay pdnhlsqvdv nelfskllkt gilklsqtds attqvsevta qpppeeeedq 1441 nedqdvpdlt nftveelkqr ydsvinrlyt giqcyscgmr fttsqtdvya dhldwhyrqn 1501 rtekdvsrkv thrrwyyslt dwiefeeiad leeraksqff ekvheevvlk tqeaakekef 1561 qsvpagpaga vesceicqeq feqywdeeee ewhlknairv dgkiyhpscy edyqntssfd 1621 ctpspsktpv enplnimlni vknelqepcd spkvkeerid tppacteesi atpseikten 1681 dtvesv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count76 = 0; count76 < 4500; count76++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgeeanddkk pttkfelere telrfeveas qsvqlelltg maeifgtelt rnkkftfdag 61 akvavftwhg csvqlsgrte vayvskdtpm llylnthtal eqmrrqaeke eergprvmvv 121 gptdvgkstv crlllnyavr lgrrptyvel dvgqgsvsip gtmgalyier padveegfsi 181 qaplvyhfgs ttpgtnikly nkitsrladv fnqrcevnrr asvsgcvint cgwvkgsgyq 241 alvhaasafe vdvvvvldqe rlynelkrdl phfvrtvllp ksggvversk dfrrecrder 301 ireyfygfrg cfyphafnvk fsdvkiykvg aptipdsclp lgmsqednql klvpvtpgrd 361 mvhhllsvst aegteenlse tsvagfivvt svdlehqvft vlspaprplp knfllimdir 421 fmdlk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atacttacct ggcaggggag ataccatgat cacgaaggtg gttttcccag ggcgaggctt 61 atccattgca ctccggatgt gctgacccct gcgatttccc caaatgtggg aaactcgact 121 gcataatttg tggtagtggg ggactgcgtt cgcgctttcc cctg';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count78 = 0; count78 < 4500; count78++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atcgcttctc ggccttttgg ctaagatcaa gtgtagtatc tgttcttatc agtttaatat 61 ctgatacgtc ctctatccga ggacaatata ttaaatggat ttttggagca gggagatgga 121 ataggagctt gctccgtcca ctccacgcat cgacctggta ttgcagtacc tccaggaacg 181 gtgcaccc';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 agctttgcgc agtggcagta tcgtagccaa tgaggtctat ccgaggcgcg attattgcta 61 attgaaaact tttcccaata ccccgccgtg acgacttgca atatagtcgg cactggcaat 121 ttttgacagt ctctacggag actg';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count80 = 0; count80 < 4500; count80++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 atactctggt ttctcttcag atcgcataaa tctttcgcct tttactaaag atttccgtgg 61 agaggaacaa ctctgagtct taacccaatt ttttgaggcc ttgctttggc aaggcta';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 gtgctcgctt cggcagcaca tatactaaaa ttggaacgat acagagaaga ttagcatggc 61 ccctgcgcaa ggatgacacg caaattcgtg aagcgttcca tatttt';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count82 = 0; count82 < 4500; count82++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 aaaaagggct tctgtcgtga gtggcacacg tagggcaact cgattgctct gcgtgcggaa 61 tcgacatcaa gagatttcgg aagcataatt ttttggtatt tgggcagctg gtgatcgttg 121 gtcccggcgc ccttt';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = ' 1 atgccttaaa cttatgagta aggaaaataa cgattcgggg tgacgcccga atcctcactg 61 ctaatgtgag acgaattttt gagcgggtaa aggtcgccct caaggtgacc cgcctacttt 121 gcgggatgcc tgggagttgc gatctgcccg';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count84 = 0; count84 < 4500; count84++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = ' 1 aaccatcctt ttcttggggt tgcgctactg tccaatgagc gcatagtgag ggcagtactg 61 ctaacgcctg aacaacacac ccgcatcaac tagagctttt gctttatttt ggtgcaattt 121 ttggaaaaat';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = '1 gtgttgtatg aaaggagaga aggttagcac tccccttgac aaggatggaa gaggccctcg 61 ggcctgacaa cacgcatacg gttaaggcat tgccacctac ttcgtggcat ctaaccatcg 121 ttttt';

rna();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count86 = 0; count86 < 4500; count86++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtqflppnll alfaprdpip ylppleklph ekhhnqpycg iapyirefed prdappptra 61 etreermerk rrekierrqq evetelkmwd phndpnaqgd afktlfvarv nydttesklr 121 refevygpik rihmvyskrs gkprgyafie yeherdmhsa ykhadgkkid grrvlvdver 181 grtvkgwrpr rlggglggtr rggadvnirh sgrddtsryd erpgpsplph rdrdrdrere 241 rrersrerdk ererrrsrsr drrrrsrsrd keerrrsrer skdkdrdrkr rssrsrerar 301 rererkeelr ggggdmaeps eagdappddg ppgelgpdgp dgpeekgrdr drerrrshrs 361 ererrrdrdr drdrdrehkr gergsergrd eargggggqd ngleglgnds rdmymesegg 421 dgylapengy lmeaape';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavpetrpnh tiyinnlnek ikkdelkksl yaifsqfgqi ldilvsrslk mrgqafvifk 61 evssatnalr smqgfpfydk pmriqyaktd sdiiakmkgt fverdrkrek rkpksqetpa 121 tkkavqggga tpvvgavqgp vpgmppmtqa primhhmpgq ppympppgmi pppglapgqi 181 ppgamppqql mpgqmppaqp lsenppnhil fltnlpeetn elmlsmlfnq fpgfkevrlv 241 pgrhdiafve fdnevqagaa rdalqgfkit qnnamkisfa kk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count88 = 0; count88 < 4500; count88++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpkfycdycd tylthdspsv rkthcsgrkh kenvkdyyqk wmeeqaqsli dkttaafqqg 61 kipptpfsap ppagamippp pslpgpprpg mmpaphmggp pmmpmmgppp pgmmpvgpap 121 gmrppmgghm pmmpgppmmr pparpmmvpt rpgmtrpdr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpagpvqavp ppppvptepk qpteeeassk edsapskpvv giiypppevr nivdktasfv 61 arngpefear irqneinnpk fnflnpndpy hayyrhkvse fkegkaqeps aaipkvmqqq 121 qqttqqqlpq kvqaqviqet ivpkepppef efiadppsis afdldvvklt aqfvarngrq 181 fltqlmqkeq rnyqfdflrp qhslfnyftk lveqytkili ppkglfsklk keaenprevl 241 dqvcyrvewa kfqererkke eeekekerva yaqidwhdfv vvetvdfqpn eqgnfppptt 301 peelgarili qeryekfges eevemevesd eeddkqekae eppsqldqdt qvqdmdegsd 361 deeegqkvpp ppetpmpppl pptpdqvivr kdydpkaskp lppapapdey lvspitgeki 421 paskmqehmr iglldprwle qrdrsirekq sddevyapgl diesslkqla errtdifgve 481 etaigkkige eeiqkpeekv twdghsgsma rtqqaaqani tlqeqieaih kakglvpedd 541 tkekigpskp neipqqpppp ssatnipssa ppitsvprpp tmpppvrttv vsavpvmprp 601 pmasvvrlpp gsviapmppi ihaprinvvp mppsappima prpppmivpt afvpappvap 661 vpapapmppv hppppmedep tskklkteds lmpeeeflrr nkgpvsikvq vpnmqdktew 721 klngqvlvft lpltdqvsvi kvkiheatgm pagkqklqye gifikdsnsl ayynmangav 781 ihlalkergg rkk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count90 = 0; count90 < 4500; count90++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdfqhrpggk tgsggvasss esnrdrrerl rqlaletidi nkdpyfmknh lgsyecklcl 61 tlhnnegsyl ahtqgkkhqt nlarraakea keapaqpape kvkvevkkfv kigrpgykvt 121 kqrdsemgqq sllfqidype iaegimprhr fmsayeqrie ppdrrwqyll maaepyetia 181 fkvpsreidk aegkfwthwn retkqfflqf hfkmekppap pslpagppgv krpppplmng 241 lpprpplpes lpppppgglp lppmpptgpa psgppgppql pppapgvhpp apvvhppasg 301 vhppapgvhp papgvhppap gvhpptsgvh ppapgvhppa pgvhppapgv hppapgvhpp 361 apgvhpppsa gvhpqapgvh paapavhpqa pgvhppapgm hpqapgvhpq ppgvhpsapg 421 vhpqppgvhp snpgvhpptp mppmlrpplp segpgnippp pptn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 metileqqrr yheekerlmd vmakemltkk stlrdqinsd hrtramqdry mevsgnlrdl 61 yddkdglrke elnaisgpne faefynrlkq ikefhrkhpn eicvpmsvef eellkarenp 121 seeaqnlvef tdeegygryl dlhdcylkyi nlkasekldy itylsifdql fdipkerkna 181 eykrylemll eylqdytdrv kplqdqnelf gkiqaefekk wengtfpgwp ketssaltha 241 gahldlsafs sweelaslgl drlksallal glkcggtlee raqrlfstkg kslesldtsl 301 faknpkskgt krdternkdi afleaqiyey veilgeqrhl thenvqrkqa rtgeereeee 361 eeqisesese deeneiiynp knlplgwdgk pipywlyklh glninyncei cgnytyrgpk 421 afqrhfaewr hahgmrclgi pntahfanvt qiedavslwa klklqkaser wqpdteeeye 481 dssgnvvnkk tyedlkrqgl l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count92 = 0; count92 < 4500; count92++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvkltaelie qaaqytnavr dreldlrgyk ipvienlgat ldqfdaidfs dneirkldgf 61 pllrrlktll vnnnricrig egldqalpcl teliltnnsl velgdldpla slksltylsi 121 lrnpvtnkkh yrlyviykvp qvrvldfqkv klkerqeaek mfkgkrgaql akdiarrskt 181 fnpgaglptd kkkggpspgd veaiknaian astlaeverl kgllqsgqip grerrsgptd 241 dgeeemeedt vtngs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdirpnhtiy innmndkikk eelkrslyal fsqfghvvdi valktmkmrg qafvifkelg 61 sstnalrqlq gfpfygkpmr iqyaktdsdi iskmrgtfad kekkkekkka ktveqtattt 121 nkkpgqgtpn santqgnstp npqvpdyppn yilflnnlpe etnemmlsml fnqfpgfkev 181 rlvpgrhdia fvefendgqa gaardalqgf kitpshamki tyakk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count94 = 0; count94 < 4500; count94++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 magtglvage vvvdalpyfd qgyeapgvre aaaalveeet rryrptknyl syltapdysa 61 fetdimrnef erlaarqpie llsmkryelp apssgqkndi tawqecvnns maqlehqavr 121 ienlelmsqh gcnawkvyne nlvhmiehaq kelqklrkhi qdlnwqrknm qltagsklre 181 mesnwvslvs knyeiertiv qleneiyqik qqhgeanken irqdf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mprimikggv wrntedeilk aavmkygknq wsriasllhr ksakqckarw yewldpsikk 61 tewsreeeek llhlaklmpt qwrtiapiig rtaaqclehy eflldkaaqr dneeettddp 121 rklkpgeidp npetkparpd pidmdedele mlseararla ntqgkkakrk arekqleear 181 rlaalqkrre lraagieiqk krkrkrgvdy naeipfekkp algfydtsee nyqaldadfr 241 klrqqdldge lrsekegrdr kkdkqhlkrk kesdlpsail qtsgvseftk krsklvlpap 301 qisdaelqev vkvgqaseia rqtaeesgit nsasstllse ynvtnnsval rtprtpasqd 361 rilqeaqnlm altnvdtplk gglntplhes dfsgvtpqrq vvqtpntvls tpfrtpsnga 421 egltprsgtt pkpvinstpg rtplrdklni npedgmadys dpsyvkqmer esrehlrlgl 481 lglpapkndf eivlpenaek eleereiddt yiedaadvda rkqairdaer vkemkrmhka 541 vqkdlprpse vnetilrpln veppltdlqk seelikkemi tmlhydllhh pyepsgnkkg 601 ktvgfgtnns ehitylehnp yekfskeelk kaqdvlvqem evvkqgmshg elsseaynqv 661 weecysqvly lpgqsrytra nlaskkdrie slekrleinr ghmtteakra akmekkmkil 721 lggyqsramg lmkqlndlwd qieqahlelr tfeelkkhed saiprrlecl kedvqrqqer 781 ekelqhryad llleketlks kf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count96 = 0; count96 < 4500; count96++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdvgellsyq pnrgtkrprd deeeeqkmrr kqtgtrergr yreeemtvve eadddkkrll 61 qiidrdgeee eeeeepldes svkkmiltfe krsyknqelr ikfpdnpekf meseldlndi 121 iqemhvvatm pdlyhllvel navqsllgll ghdntdvsia vvdllqeltd idtlheseeg 181 aevlidalvd gqvvallvqn lerldesvke eadgvhntla ivenmaefrp emctegaqqg 241 llqwllkrlk akmpfdankl ycsevlaill qdndenrell geldgidvll qqlsvfkrhn 301 pstaeeqemm enlfdslcsc lmlssnrerf lkgeglqlmn lmlrekkisr ssalkvldha 361 migpegtdnc hkfvdilglr tifplfmksp rkikkvgtte keheehvcsi lasllrnlrg 421 qqrtrllnkf tendsekvdr lmelhfkylg amqvadkkie gekhdmvrrg eiidndteee 481 fylrrldagl fvlqhicyim aeicnanvpq irqrvhqiln mrgssikivr hiikeyaeni 541 gdgrspefre neqkrilgll enf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mttaarptfe parggrgkge gdlsqlskqy ssrdlpshtk ikyrqttqda peevrnrdfr 61 releereraa areknrdrpt rehttsssvs kkprldqipa anldaddplt deededfeee 121 sddddtaall aelekikker aeeqarkeqe qkaeeerirm enilsgnpll nltgpsqpqa 181 nfkvkrrwdd dvvfkncakg vddqkkdkrf vndtlrsefh kkfmekyik';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count98 = 0; count98 < 4500; count98++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mveevqkhsv htlvfrslkr thdmfvadng kpvpldeesh krkmaiklrn eygpvlhmpt 61 skenlkekgp qnatdsyvhk qypanqgqev eyfvagthpy ppgpgvalta dtkiqrmpse 121 saaqslaval plqtkadanr tapsgseyrh pgasdrpqpt amnsivmetg ntknsalmak 181 kaptmpkpqw hppwklyrvi sghlgwvrci avepgnqwfv tgsadrtiki wdlasgklkl 241 sltghistvr gvivstrspy lfscgedkqv kcwdleynkv irhyhghlsa vygldlhpti 301 dvlvtcsrds tariwdvrtk asvhtlsght navatvrcqa aepqiitgsh dttirlwdlv 361 agktrvtltn hkksvravvl hprhytfasg spdnikqwkf pdgsfiqnls ghnaiintlt 421 vnsdgvlvsg adngtmhlwd wrtgynfqrv haavqpgsld sesgifacaf dqsesrllta 481 eadktikvyr eddtateeth pvswkpeiik rkrf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslicsisne vpehpcvspv snhvyerrli ekyiaengtd pinnqplsee qlidikvahp 61 irpkppsats ipailkalqd ewdavmlhsf tlrqqlqttr qelshalyqh daacrviarl 121 tkevtaarea latlkpqagl ivpqavpssq psvvgagepm dlgelvgmtp eiiqklqdka 181 tvltterkkr gktvpeelvk peelskyrqv ashvglhsas ipgilaldlc psdtnkiltg 241 gadknvvvfd ksseqilatl kghtkkvtsv vfhpsqdlvf saspdatiri wsvpnascvq 301 vvrahesavt glslhatgdy llsssddqyw afsdiqtgrv ltkvtdetsg csltcaqfhp 361 dglifgtgtm dsqikiwdlk ertnvanfpg hsgpitsiaf sengyylata addssvklwd 421 lrklknfktl qldnnfevks lifdqsgtyl alggtdvqiy ickqwteilh ftehsglttg 481 vafghhakfi astgmdrslk fysl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count100 = 0; count100 < 4500; count100++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgsskkhrge keaagttaaa gtggateqpp rhrehkkhkh rsggsggsgg errkrsrerg 61 gergsgrrga eaearssthg rersqaepse rrvkrekrdd gyeaaasskt ssgdasslsi 121 eetnklrakl glkplevnai kkeagtkeep vtadvinpma lrqreelrek laaakekrll 181 nqklgkiktl geddpwlddt aawiersrql qkekdlaekr aklleemdqe fgvstlveee 241 fgqrrqdlys ardlqgltve haidsfrege tmiltlkdkg vlqeeedvlv nvnlvdkera 301 eknvelrkkk pdylpyaede svddlaqqkp rsilskydee legerphsfr leqggtadgl 361 rereleeira klrlqaqsls tvgprlasey ltpeemvtfk ktkrrvkkir kkekevvvra 421 ddllplgdqt qdgdfgsrlr grgrrrvsev eeekepvpqp lpsddtrven mdisdeeegg 481 apppgspqvl eedeaelelq kqlekgrrlr qlqqlqqlrd sgekvveivk klesrqrgwe 541 ededperkga ivfnatsefc rtlgeiptyg lagnreeqee lmdferdeer sanggsesdg 601 eenigwstvn ldeekqqqdf sassttilde epivnrglaa alllcqnkgl lettvqkvar 661 vkapnkslps avyciedkma iddkysrree yrgftqdfke kdgykpdvki eyvdetgrkl 721 tpkeafrqls hrfhgkgsgk mkterrmkkl deeallkkms ssdtplgtva llqekqkaqk 781 tpyivlsgsg ksmnantitk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgsskkhrge keaagttaaa gtggateqpp rhrehkkhkh rsggsggsgg errkrsrerg 61 gergsgrrga eaearssthg rersqaepse rrvkrekrdd gyeaaasskt ssgdasslsi 121 eetnklrakl glkplevnai kkeagtkeep vtadvinpma lrqreelrek laaakekrll 181 nqklgkiktl geddpwlddt aawiersrql qkekdlaekr aklleemdqe fgvstlveee 241 fgqrrqdlys ardlqgltve haidsfrege tmiltlkdkg vlqeeedvlv nvnlvdkera 301 eknvelrkkk pdylpyaede svddlaqqkp rsilskydee legerphsfr leqggtadgl 361 rereleeira klrlqaqsls tvgprlasey ltpeemvtfk ktkrrvkkir kkekevvvra 421 ddllplgdqt qdgdfgsrlr grgrrrvsev eeekepvpqp lpsddtrven mdisdeeegg 481 apppgspqvl eedeaelelq kqlekgrrlr qlqqlqqlrd sgekvveivk klesrqrgwe 541 ededperkga ivfnatsefc rtlgeiptyg lagnreeqee lmdferdeer sanggsesdg 601 eenigwstvn ldeekqqqdf sassttilde epivnrglaa alllcqnkgl lettvqkvar 661 vkapnkslps avyciedkma iddkysrree yrgftqdfke kdgykpdvki eyvdetgrkl 721 tpkeafrqls hrfhgkgsgk mkterrmkkl deeallkkms ssdtplgtva llqekqkaqk 781 tpyivlsgsg ksmnantitk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

unused\_variable = 5;

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count102 = 0; count102 < 4500; count102++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madrrrqras qdtedeesga sgsdsggspl rgggscsgsa ggggsgslps qrggrtgalh 61 lrrvesggak saeesecese dgiegdavls dyesaedseg eegeyseeen skvelksean 121 davnsstkee kgeekpdtks tvtgerqsgd gqestepven kvgkkgpkhl dddedrknpa 181 yiprkglffe hdlrgqtqee evrpkgrqrk lwkdegrweh dkfredeqap ksrqelialy 241 gydirsahnp ddikprrirk prygsppqrd pnwngerlnk shrhqglggt lpprtfinrn 301 aagtgrmsap rnysrsggfk egragfrpve aggqhggrsg etvkheisyr srrleqtsvr 361 dpspeadapv lgspekeeaa seppaaapda appppdrpie kksysrarrt rtkvgdavkl 421 aeevppppeg lipappvpet tptpptktgt weapvdssts gleqdvaqln iaeqnwspgq 481 psflqprelr gmpnhihmga gpppqfnrme emgvqggrak ryssqrqrpv peppappvhi 541 simeghyydp lqfqgpiyth gdspaplppq gmlvqpgmnl phpglhphqt paplpnpgly 601 pppvsmspgq pppqqllapt yfsapgvmnf gnpsypyapg alpppppphl ypntqapsqv 661 yggvtyynpa qqqvqpkpsp prrtpqpvti kppppevvsr gss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mattatmats gsarkrllke edmtkvefet seevdvtptf dtmglredll rgiyaygfek 61 psaiqqraik qiikgrdvia qsqsgtgkta tfsisvlqcl diqvretqal ilaptrelav 121 qiqkgllalg dymnvqchac iggtnvgedi rkldygqhvv agtpgrvfdm irrrslrtra 181 ikmlvldead emlnkgfkeq iydvyrylpp atqvvlisat lpheilemtn kfmtdpiril 241 vkrdeltleg ikqffvaver eewkfdtlcd lydtltitqa vifcntkrkv dwltekmrea 301 nftvssmhgd mpqkeresim kefrsgasrv listdvwarg ldvpqvslii nydlpnnrel 361 yihrigrsgr ygrkgvainf vknddirilr dieqyystqi dempmnvadl i';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count104 = 0; count104 < 4500; count104++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mesdfylryy vghkgkfghe flefefrpdg klryannsny kndvmirkea yvhksvmeel 61 kriiddseit keddalwppp drvgrqelei vigdehisft tskigslidv nqskdpeglr 121 vfyylvqdlk clvfsliglh fkikpi';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavasdfylr yyvghkgkfg heflefefrp dgklryanns nykndvmirk eayvhksvme 61 elkriiddse itkeddalwp ppdrvgrqel eivigdehis fttskigsli dvnqskdpeg 121 lrvfyylvqd lkclvfslig lhfkikpi';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count106 = 0; count106 < 4500; count106++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madvldlhea ggedfamded gdesihklke kakkrkgrgf gseegsrarm redydsveqd 61 gdepgpqrsv egwilfvtgv heeateedih dkfaeygeik nihlnldrrt gylkgytlve 121 yetykeaqaa meglngqdlm gqpisvdwcf vrgppkgkrr ggrrrsrspd rrrr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnympgtasl iedidkkhlv llrdgrtlig flrsidqfan lvlhqtveri hvgkkygdip 61 rgifvvrgen vvllgeidle kesdtplqqv sieeileeqr veqqtkleae klkvqalkdr 121 glsipradtl dey';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count108 = 0; count108 < 4500; count108++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlfysffksl vgkdvvvelk ndlsicgtlh svdqylnikl tdisvtdpek yphmlsvknc 61 firgsvvryv qlpadevdtq llqdaarkea lqqkq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count109 = 0; count109 < 4500; count109++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maddvdqqqt tntveepldl irlslderiy vkmrndrelr grlhaydqhl nmilgdveet 61 vttieideet yeeiykstkr nipmlfvrgd gvvlvapplr vg';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count110 = 0; count110 < 4500; count110++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlplsllkta qnhpmlvelk ngetynghlv scdnwmninl revictsrdg dkfwrmpecy 61 irgstikylr ipdeiidmvk eevvakgrgr gglqqqkqqk grgmggagrg vfggrgrggi 121 pgtgrgqpek kpgrqagkq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count111 = 0; count111 < 4500; count111++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maanattnps qllplelvdk cigsrihivm ksdkeivgtl lgfddfvnmv ledvtefeit 61 pegrritkld qillngnnit mlvpggegpe v';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count112 = 0; count112 < 4500; count112++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslrkqtpsd flkqiigrpv vvklnsgvdy rgvlacldgy mnialeqtee yvngqlknky 61 gdafirgnnv lyistqkrrm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count113 = 0; count113 < 4500; count113++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madkekkkke sildlskyid ktirvkfqgg reasgilkgf dpllnlvldg tieymrdpdd 61 qykltedtrq lglvvcrgts vvlicpqdgm eaipnpfiqq qda';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count114 = 0; count114 < 4500; count114++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtsalenyin rtvavitsdg rmivgtlkgf dqtinlilde shervfsssq gveqvvlgly 61 ivrgdnvavi geideetdsa ldlgniraep lnsvah';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count115 = 0; count115 < 4500; count115++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavshsvker tisensliil lqglqgrvtt vdlrdesvah gridnvdafm nirlakvtyt 61 drwghqvkld dlfvtgrnvr yvhipddvni tstieqqlqi ihrvrnfggk gqgrwefppk 121 nck';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count116 = 0; count116 < 4500; count116++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 meerergars agagsparpp sprldvssds fdpllalyap rlppipypna pcfnnvaeye 61 sflrtgvrgg grgrgrarga aagsgvpaap gpsgrtrrrp dapapdperi qrlrrlmvak 121 eegdgaagag rrgpgrsrka prnvltrmpl hegsplgelh rciregvkvn vhirtfkglr 181 gvctgflvaf dkfwnmaltd vdetyrkpvl gkayerdssl tltrlfdrlk lqdsskkead 241 sksavedstl srysqtstwk lasvwgradt grgshkrsrs vpsslqasar eesrselsgr 301 ttrtdgssvg gtfsrattls rgqsrkkkrk pkvdyqqvft rhinqifirg envllvhlaq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count117 = 0; count117 < 4500; count117++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maappgeyfs vgsqvscrtc qeqrlqgevv afdyqskmla lkcpsssgkp nhadillinl 61 qyvseveiin drtetpppla slnvsklask artekeekls qayaisagvs legqqlfqti 121 hktikdckwq eknivvmeev vitppyqven ckgkegsals hvrkirqsla lsptlecsga 181 isahcnlrlp gssdspasas rvpgttgvch htrlemgfhh vgqaglellt';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count118 = 0; count118 < 4500; count118++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msggtpyigs kisliskaei ryegilytid tenstvalak vrsfgtedrp tdrpipprde 61 vfeyiifrgs dikdltvcep pkpqcslpqd paivqsslgs stssfqsmgs ygpfgrmpty 121 sqfspsslvg qqfgavgvag ssltsfgtet snsgtlpqss avgsaftqdt rslktqlsqg 181 rsspqldplr ksptmeqavq tasahlpapa avgrrspvst rplpsasqka genqehrrae 241 vhkvsrpene qlrndnkrqv apgapsaprr grgghrggrg rfgirrdgpm kfekdfdfes 301 anaqfnkeei drefhnklkl kedklekqek pvngedkgds gvdtqnsegn adeedplgpn 361 cyydktksff dniscddnre rrptwaeerr lnaetfgipl rpnrgrggyr grgglgfrgg 421 rgrgggrggt ftaprgfrgg frggrggref adfeyrktta fgp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count119 = 0; count119 < 4500; count119++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSGSSGTPYLGSKISLISKAQIRYEGILYTIDTDNSTVALAKVR SFGTEDRPTDRPAPPREEIYEYIIFRGSDIKDITVCEPPKAQHTLPQDPAIVQSSLGS ASASPFQPHVPYSPFRGMAPYGPLAASSLLSQQYAASLGLGAGFPSIPVGKSPMVEQA VQTGSADNLNAKKLLPGKGTTGTQLNGRQAQPSSKTASDVVQPAAVQAQGQVNDENRR PQRRRSGNRRTRNRSRGQNRPTNVKENTIKFEGDFDFESANAQFNREELDKEFKKKLN FKDDKAEKGEEKDLAVVTQSAEAPAEEDLLGPNCYYDKSKSFFDNISSELKTSSRRTT WAEERKLNTETFGVSGRFLRGRSSRGGFRGGRGNGTTRRNPTSHRAGTGRV';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count120 = 0; count120 < 4500; count120++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpagpvqavp ppppvptepk qpteeeassk edsapskpvv giiypppevr nivdktasfv 61 arngpefear irqneinnpk fnflnpndpy hayyrhkvse fkegkaqeps aaipkvmqqq 121 qqttqqqlpq kvqaqviqet ivpkepppef efiadppsis afdldvvklt aqfvarngrq 181 fltqlmqkeq rnyqfdflrp qhslfnyftk lveqytkili ppkglfsklk keaenprevl 241 dqvcyrvewa kfqererkke eeekekerva yaqidwhdfv vvetvdfqpn eqgnfppptt 301 peelgarili qeryekfges eevemevesd eeddkqekae eppsqldqdt qvqdmdegsd 361 deeegqkvpp ppetpmpppl pptpdqvivr kdydpkaskp lppapapdey lvspitgeki 421 paskmqehmr iglldprwle qrdrsirekq sddevyapgl diesslkqla errtdifgve 481 etaigkkige eeiqkpeekv twdghsgsma rtqqaaqani tlqeqieaih kakglvpedd 541 tkekigpskp neipqqpppp ssatnipssa ppitsvprpp tmpppvrttv vsavpvmprp 601 pmasvvrlpp gsviapmppi ihaprinvvp mppsappima prpppmivpt afvpappvap 661 vpapapmppv hppppmedep tskklkteds lmpeeeflrr nkgpvsikvq vpnmqdktew 721 klngqvlvft lpltdqvsvi kvkiheatgm pagkqklqye gifikdsnsl ayynmangav 781 ihlalkergg rkk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count121 = 0; count121 < 4500; count121++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdfqhrpggk tgsggvasss esnrdrrerl rqlaletidi nkdpyfmknh lgsyecklcl 61 tlhnnegsyl ahtqgkkhqt nlarraakea keapaqpape kvkvevkkfv kigrpgykvt 121 kqrdsemgqq sllfqidype iaegimprhr fmsayeqrie ppdrrwqyll maaepyetia 181 fkvpsreidk aegkfwthwn retkqfflqf hfkmekppap pslpagppgv krpppplmng 241 lpprpplpes lpppppgglp lppmpptgpa psgppgppql pppapgvhpp apvvhppasg 301 vhppapgvhp papgvhppap gvhpptsgvh ppapgvhppa pgvhppapgv hppapgvhpp 361 apgvhpppsa gvhpqapgvh paapavhpqa pgvhppapgm hpqapgvhpq ppgvhpsapg 421 vhpqppgvhp snpgvhpptp mppmlrpplp segpgnippp pptn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count122 = 0; count122 < 4500; count122++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 metileqqrr yheekerlmd vmakemltkk stlrdqinsd hrtramqdry mevsgnlrdl 61 yddkdglrke elnaisgpne faefynrlkq ikefhrkhpn eicvpmsvef eellkarenp 121 seeaqnlvef tdeegygryl dlhdcylkyi nlkasekldy itylsifdql fdipkerkna 181 eykrylemll eylqdytdrv kplqdqnelf gkiqaefekk wengtfpgwp ketssaltha 241 gahldlsafs sweelaslgl drlksallal glkcggtlee raqrlfstkg kslesldtsl 301 faknpkskgt krdternkdi afleaqiyey veilgeqrhl thenvqrkqa rtgeereeee 361 eeqisesese deeneiiynp knlplgwdgk pipywlyklh glninyncei cgnytyrgpk 421 afqrhfaewr hahgmrclgi pntahfanvt qiedavslwa klklqkaser wqpdteeeye 481 dssgnvvnkk tyedlkrqgl l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count123 = 0; count123 < 4500; count123++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 makhhpdlif crkqagvaig rlcekcdgkc vicdsyvrpc tlvricdecn ygsyqgrcvi 61 cggpgvsday yckectiqek drdgcpkivn lgssktdlfy erkkygfkkr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count124 = 0; count124 < 4500; count124++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 makiakthed ieaqireiqg kkaaldeaqg vgldstgyyd qeiyggsdsr fagyvtsiaa 61 teledddddy ssstsllgqk kpgyhapval lndipqsteq ydpfaehrpp kiadredeyk 121 khrrtmiisp erldpfadgg ktpdpkmnar tymdvmreqh ltkeereirq qlaekakage 181 lkvvngaaas qppskrkrrw dqtadqtpga tpkklsswdq aetpghtpsl rwdetpgrak 241 gsetpgatpg skiwdptpsh tpagaatpgr gdtpghatpg hggatssark nrwdetpkte 301 rdtpghgsgw aetprtdrgg dsigetptpg askrksrwde tpasqmggst pvltpgktpi 361 gtpamnmatp tpghimsmtp eqlqawrwer eidernrpls deeldamfpe gykvlpppag 421 yvpirtpark ltatptplgg mtgfhmqted rtmksvndqp sgnlpflkpd diqyfdkllv 481 dvdestlspe eqkerkimkl llkikngtpp mrkaalrqit dkarefgagp lfnqilpllm 541 sptledqerh llvkvidril yklddlvrpy vhkilvviep llidedyyar vegreiisnl 601 akaaglatmi stmrpdidnm deyvrnttar afavvasalg ipsllpflka vckskkswqa 661 rhtgikivqq iailmgcail phlrslveii ehglvdeqqk vrtisalaia alaeaatpyg 721 iesfdsvlkp lwkgirqhrg kglaaflkai gyliplmdae yanyytrevm lilirefqsp 781 deemkkivlk vvkqccgtdg veanyiktei lppffkhfwq hrmaldrrny rqlvdttvel 841 ankvgaaeii srivddlkde aeqyrkmvme tiekimgnlg aadidhklee qlidgilyaf 901 qeqttedsvm lngfgtvvna lgkrvkpylp qicgtvlwrl nnksakvrqq aadlisrtav 961 vmktcqeekl mghlgvvlye ylgeeypevl gsilgalkai vnvigmhkmt ppikdllprl 1021 tpilknrhek vqencidlvg riadrgaeyv sarewmricf ellellkahk kairratvnt 1081 fgyiakaigp hdvlatllnn lkvqerqnrv cttvaiaiva etcspftvlp almneyrvpe 1141 lnvqngvlks lsflfeyige mgkdyiyavt plledalmdr dlvhrqtasa vvqhmslgvy 1201 gfgcedslnh llnyvwpnvf etsphviqav mgaleglrva igpcrmlqyc lqglfhpark 1261 vrdvywkiyn siyigsqdal iahypriynd dkntyiryel dyil';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count125 = 0; count125 < 4500; count125++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matehpeppk aelqlppppp pghygawaaq elqaklaeig apiqgnreel verlqsytrq 61 tgivlnrpvl rgedgdkaap ppmsaqlpgi pmpppplglp plqppppppp pppglglgfp 121 mahppnlgpp pplrvgepva lseeerlkla qqqaallmqq eerakqqgdh slkehelleq 181 qkraavlleq erqqeiakmg tpvprppqdm gqigvrtplg prvaapvgpv gptptvlpmg 241 apvprprgpp pppgdenrem ddpsvgpkip qalekilqlk esrqeemnsq qeeeemetda 301 rsslgqsase teedtvsvsk keknrkrrnr kkkkkpqrvr gvssessgdr ekdstrsrgs 361 dspaadveie yvteepeiye pnfiffkrif eafkltddvk kekekepekl dklensaapk 421 kkgfeeehkd sdddssddeq ekkpeapkls kkklrrmnrf tvaelkqlva rpdvvemhdv 481 taqdpkllvh lkatrnsvpv prhwcfkrky lqgkrgiekp pfelpdfikr tgiqemreal 541 qekeeqktmk skmrekvrpk mgkididyqk lhdaffkwqt kpkltihgdl yyegkefetr 601 lkekkpgdls delrislgmp vgpnahkvpp pwliamqryg pppsypnlki pglnspipes 661 csfgyhaggw gkppvdetgk plygdvfgtn aaefqtktee eeidrtpwge lepsdeesse 721 eeeeeesded kpdetgfitp adsglitpgg fssvpagmet pelielrkkk ieeamdgset 781 pqlftvlpek rtatvggamm gsthiydmst vmsrkgpape lqgvevalap eeleldpmam 841 tqkyeehvre qqaqvekedf sdmvaehaak qkqkkrkaqp qdsrggskky kefkf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count126 = 0; count126 < 4500; count126++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mflynltlqr atgisfaihg nfsgtkqqei vvsrgkilel lrpdpntgkv htlltvevfg 61 virslmafrl tggtkdyivv gsdsgrivil eyqpsknmfe kihqetfgks gcrrivpgqf 121 lavdpkgrav misaiekqkl vyilnrdaaa rltissplea hkantlvyhv vgvdvgfenp 181 mfaclemdye eadndptgea aantqqtltf yeldlglnhv vrkyseplee hgnflitvpg 241 gsdgpsgvli csenyitykn fgdqpdircp iprrrndldd pergmifvcs athktksmff 301 flaqteqgdi fkitletded mvteirlkyf dtvpvaaamc vlktgflfva sefgnhylyq 361 iahlgdddee pefssample egdtfffqpr plknlvlvde ldslspilfc qiadlanedt 421 pqlyvacgrg prsslrvlrh glevsemavs elpgnpnavw tvrrhiedef dayiivsfvn 481 atlvlsiget veevtdsgfl gttptlscsl lgddalvqvy pdgirhirad krvnewktpg 541 kktivkcavn qrqvvialtg gelvyfemdp sgqlneyter kemsadvvcm slanvppgeq 601 rsrflavglv dntvriisld psdclqplsm qalpaqpesl civemggtek qdelgergsi 661 gflylniglq ngvllrtvld pvtgdlsdtr trylgsrpvk lfrvrmqgqe avlamssrsw 721 lsysyqsrfh ltplsyetle fasgfaseqc pegivaistn tlrilalekl gavfnqvafp 781 lqytprkfvi hpesnnliii etdhnaytea tkaqrkqqma eemveaaged erelaaemaa 841 aflnenlpes ifgapkagng qwasvirvmn piqgntldlv qleqneaafs vavcrfsntg 901 edwyvlvgva kdlilnprsv aggfvytykl vnngeklefl hktpveevpa aiapfqgrvl 961 igvgkllrvy dlgkkkllrk cenkhianyi sgiqtighrv ivsdvqesfi wvrykrnenq 1021 liifaddtyp rwvttaslld ydtvagadkf gnicvvrlpp ntndevdedp tgnkalwdrg 1081 llngasqkae vimnyhvget vlslqkttli pggseslvyt tlsggigilv pftshedhdf 1141 fqhvemhlrs ehpplcgrdh lsfrsyyfpv knvidgdlce qfnsmepnkq knvseeldrt 1201 ppevskkled irtryaf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count127 = 0; count127 < 4500; count127++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maagpisern qdatvyvggl dekvsepllw elflqagpvv nthmpkdrvt gqhqgygfve 61 flseedadya ikimnmikly gkpirvnkas ahnknldvga nifignldpe idekllydtf 121 safgvilqtp kimrdpdtgn skgyafinfa sfdasdaaie amngqylcnr pitvsyafkk 181 dskgerhgsa aerllaaqnp lsqadrphql fadappppsa pnpvvsslgs glpppgmppp 241 gsfpppvppp galppgippa mppppmppga aghgppsagt pgaghpghgh shphpfppgg 301 mphpgmsqmq lahhgphglg hphagppgsg gqppprpppg mphpgpppmg mpprgppfgs 361 pmghpgpmpp hgmrgppplm pphgytgppr pppygyqrgp lppprptprp pvpprgplrg 421 plpq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count128 = 0; count128 < 4500; count128++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtdrytihsq lehlqskyig tghadttkwe wlvnqhrdsy csymghfdll nyfaiaenes 61 karvrfnlme kmlqpcgppa dkpeen';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count129 = 0; count129 < 4500; count129++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mamqaakran irlppevnri lyirnlpyki taeemydifg kygpirqirv gntpetrgta 61 yvvyedifda knacdhlsgf nvcnrylvvl yynanrafqk mdtkkkeeql kllkekygin 121 tdppk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count130 = 0; count130 < 4500; count130++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtvgksskml qhidyrmrci lqdgrifigt fkafdkhmnl ilcdcdefrk ikpknskqae 61 reekrvlglv llrgenlvsm tvegpppkdt giarvplaga aggpgigraa grgipagvpm 121 pqapaglagp vrgvggpsqq vmtpqgrgtv aaaaaaatas iagaptqypp grggppppmg 181 rgapppgmmg pppgmrppmg ppmgippgrg tpmgmpppgm rppppgmrgl l';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count131 = 0; count131 < 4500; count131++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mklvrflmkl shetvtielk ngtqvhgtit gvdvsmnthl kavkmtlknr epvqletlsi 61 rgnniryfil pdslpldtll vdvepkvksk kreavagrgr grgrgrgrgr grgrggprr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count132 = 0; count132 < 4500; count132++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msllnkpkse mtpeelqkre eeefntgpls vltqsvknnt qvlincrnnk kllgrvkafd 61 rhcnmvlenv kemwtevpks gkgkkkskpv nkdryiskmf lrgdsvivvl rnpliagk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count133 = 0; count133 < 4500; count133++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msigvpikvl heaeghivtc etntgevyrg klieaednmn cqmsnitvty rdgrvaqleq 61 vyirgskirf lilpdmlkna pmlksmknkn qgsgagrgka ailkaqvaar grgrgmgrgn 121 ifqkrr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count134 = 0; count134 < 4500; count134++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mayrgqgqkv qkvmvqpinl ifrylqnrsr iqvwlyeqvn mriegciigf deymnlvldd 61 aeeihsktks rkqlgrimlk gdnitllqsv sn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count135 = 0; count135 < 4500; count135++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslplnpkpf lngltgkpvm vklkwgmeyk gylvsvdgym nmqlanteey idgalsghlg 61 evlircnnvl yirgveeeee dgemre';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count136 = 0; count136 < 4500; count136++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskahppelk kfmdkklslk lnggrhvqgi lrgfdpfmnl videcvemat sgqqnnigmv 61 virgnsiiml ealerv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count137 = 0; count137 < 4500; count137++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmdsqtvgks skmlqhidyr mrcilqdgri figtfkafdk hmnlilcdcd efrkikpkna 61 kqpereekrv lglvllrgen lvsmtvegpp pkdtgiarvp lagaaggpgv graagrgvpa 121 gvpipqapag lagpvrgvgg psqqvmtpqg rgtvaaaava atasiagapt qyppgrgtpp 181 ppvgratppp gimapppgmr ppmgppiglp pargtpigmp ppgmrppppg irgppppgmr 241 pprp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count138 = 0; count138 < 4500; count138++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maqfggqknp pwatqftata vsqpaalgvq qpsllgaspt iytqqtalaa aglttqtpan 61 yqltqtaalq qqaaaaaaal qqqysqpqqa lysvqqqlqq pqqtlltqpa valptslsls 121 tpqptaqitv syptprssqq qtqpqkqrvf tgvvtklhdt fgfvdedvff qlsavkgktp 181 qvgdrvlvea tynpnmpfkw naqriqtlpn qnqsqtqpll ktppavlqpi apqttfgvqt 241 qpqpqsllqa qisaasitpl lqtqpqpllq qpqqkagllq ppvrivsqpq parrldppsr 301 fsgrndrgdq vpnrkddrsr erererrrsr erspqrkrsr ersprrerer sprrvrrvvp 361 rytvqfskfs ldcpscdmme lrrryqnlyi psdffdaqft wvdafplsrp fqlgnycnfy 421 vmhrevesle knmaildppd adhlysakvm lmaspsmedl yhkscalaed pqelrdgfqh 481 parlvkflvg mkgkdeamai gghwspsldg pdpekdpsvl iktairccka ltgidlsvct 541 qwyrfaeiry hrpeethkgr tvpahvetvv lffpdvwhcl ptrsewetls rgykqqlvek 601 lqgerkeadg eqdeeekddg eakeistpth wskldpktmk vndlrkeles ralsskglks 661 qliarltkql kveeqkeeqk eleksekeed edddrksedd keeeerkrqe eierqrrerr 721 yilpdepaii vhpnwaaksg kfdcsimsls vlldyrledn kehsfevslf aelfnemlqr 781 dfgvriyksl lslpekedkk ekdkkskkde rkdkkeerdd etdepkpkrr ksgddkdkke 841 drderkkedk rkddskddde teednnqdey dpmeaeeaed eeddrdeeem tkrddkrdin 901 ryckerpskd kekektqmit inrdllmafv yfdqshcgyl lekdleeily tlglhlsraq 961 vkkllnkvvl rescfyrklt dtskdeenhe eseslqedml gnrlllptpt vkqeskdvee 1021 nvglivynga mvdvgsllqk leksekvrae veqklqllee ktdedektil nlensnksls 1081 gelrevkkdl sqlqenlkis enmnlqfenq mnktirnlst vmdeihtvlk kdnvknedkd 1141 qkskengasv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count139 = 0; count139 < 4500; count139++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 memplppddq elrnvidkla qfvarngpef ekmtmekqkd npkfsflfgg efysyykckl 61 aleqqqlick qqtpelepaa tmpplpqppl apaapippaq gapsmdeliq qsqwnlqqqe 121 qhllalrqeq vtaavahave qqmqklleet qldmnefdnl lqpiidtctk daisagknwm 181 fsnakspphc elmaghlrnr itadgahfel rlhliylind vlhhcqrkqa rellaalqkv 241 vvpiyctsfl aveedkqqki arllqlwekn gyfddsiiqq lqspalglgq yqatlineys 301 svvqpvqlaf qqqiqtlktq heefvtslaq qqqqqqqqqq qlqmpqmeae vkatppppap 361 ppapapapai ppttqpddsk ppiqmpgsse yeapggvqdp aaagprgpgp hdqippnkpp 421 wfdqphpvap wgqqqppeqp pyphhqggpp hcppwnnshe gmwgeqrgdp gwngqrdapw 481 nnqpdaawns qfegpwnsqh eqppwgggqr eppfrmqrpp hfrgpfpphq qhpqfnqpph 541 phnfnrfppr fmqddfpprh pferppyphr fdypqgdfpa emgpphhhpg hrmphpgine 601 hppwagpqhp dfgppphgfn gqpphmrrqg pphinhddps lvpnvpyfdl paglmaplvk 661 ledheykpld pkdirlpppm ppserllaav eafysppshd rprnsegweq nglyeffrak 721 mrarrrkgqe krnsgpsrsr srsksrgrss srsnsrssks sgsysrsrsr scsrsysrsr 781 srsrsrsrss rsrsrsqsrs rsksyspgrr rrsrsrsptp pssaglgsns appipdsrlg 841 eenkghqmlv kmgwsgsggl gakeqgiqdp ikggdvrdkw dqykgvgval ddpyenyrrn 901 ksysfiarmk ardeck';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count140 = 0; count140 < 4500; count140++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 matpaglerw vqdelhsvlg lserhvaqfl igtaqrctsa eefvqrlrdt dtldlsgpar 61 dfalrlwnkv prkavvekpa raaerearal leknrsyrll edseesseet vsragsslqk 121 krkkrkhlrk kreeeeeeea sekgkkktgg skqqtekpes edewertere rlqdleerda 181 faervrqrdk drtrnvlers dkkayeeaqk rlkmaeedrk amvpelrkks rreylakrer 241 ekledleael adeeflfgdv elsrherqel kykrrvrdla reyraageqe kleatnryhm 301 pketrgqpar avdlveeesg apgeeqrrwe earlgaaslk fgardaasqe pkyqlvleee 361 etiefvratq lqgdeepsap ptstqaqqke siqavrrslp vfpfreella aianhqvlii 421 egetgsgktt qipqylfeeg ytnkgmkiac tqprrvaams vaarvaremg vklgnevgys 481 irfedctser tvlrymtdgm llreflsepd lasysvvmvd eahertlhtd ilfglikdva 541 rfrpelkvlv asatmdtarf stffddapvf ripgrrfpvd ifytkapead yleacvvsvl 601 qihvtqppgd ilvfltgqee ieaacemlqd rcrrlgskir ellvlpiyan lpsdmqarif 661 qptppgarkv vvatniaets ltiegiiyvl dpgfckqksy nprtgmeslt vtpcskasan 721 qragragrva agkcfrlyta wayqheleet tvpeiqrtsl gnvvlllksl gihdlmhfdf 781 ldpppyetll laleqlyalg alnhlgeltt sgrkmaelpv dpmlskmila sekyscseei 841 ltvaamlsvn nsifyrpkdk vvhadnarvn fflpggdhlv llnvytqwae sgyssqwcye 901 nfvqfrsmrr ardvreqleg llervevgls scqgdyirvr kaitagyfyh tarltrsgyr 961 tvkqqqtvfi hpnsslfeqq prwllyhelv lttkefmrqv leiesswlle vaphyykake 1021 ledphakkmp kkigktreel g';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count141 = 0; count141 < 4500; count141++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mskrhrldlg edypsgkkra gtdgkdrdrd rdredrskdr drerdrgdre rerekekeke 61 lrastnamli saglpplkas hsahsthsah sthsthsahs thaghaghts lpqcinpftn 121 lphtpryydi lkkrlqlpvw eykdrftdil vrhqsfvlvg etgsgkttqi pqwcveymrs 181 lpgpkrgvac tqprrvaams vaqrvademd vmlgqevgys irfedcssak tilkymtdgm 241 llreamndpl lerygviild eahertlatd ilmgvlkevv rqrsdlkviv msatldagkf 301 qiyfdncpll tipgrthpve ifytpeperd yleaairtvi qihmceeeeg dlllfltgqe 361 eideackrik revddlgpev gdikiiplys tlppqqqqri feppppkkqn gaigrkvvvs 421 tniaetslti dgvvfvidpg fakqkvynpr irvesllvta iskasaqqra gragrtrpgk 481 cfrlytekay ktemqdntyp eilrsnlgsv vlqlkklgid dlvhfdfmdp papetlmral 541 ellnylaaln ddgdltelgs mmaefpldpq lakmviascd yncsnevlsi tamlsvpqcf 601 vrpteakkaa deakmrfahi dgdhltllnv yhafkqnhes vqwcydnfin yrslmsadnv 661 rqqlsrimdr fnlprrstdf tsrdyyinir kalvtgyfmq vahlertghy ltvkdnqvvq 721 lhpstvldhk pewvlynefv lttknyirtc tdikpewlvk iapqyydmsn fpqceakrql 781 driiaklqsk eysqy';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count142 = 0; count142 < 4500; count142++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mseageeqpm ettgatengh eavpegespa gagtgaaaga ggataappsg nqngaegdqi 61 naskneedag kmfvgglswd tskkdlkdyf tkfgevvdct ikmdpntgrs rgfgfilfkd 121 aasvekvldq kehrldgrvi dpkkamamkk dpvkkifvgg lnpeateeki reyfgefgei 181 eaielpmdpk lnkrrgfvfi tfkeeepvkk vlekkfhtvs gskceikvaq pkevyqqqqy 241 gsggrgnrnr gnrgsggggg gggqsqswnq gygnywnqgy gyqqgygpgy ggydyspygy 301 ygygpgydys qgstnygksq rrgghqnnyk py';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count143 = 0; count143 < 4500; count143++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msksespkep eqlrklfigg lsfettdesl rshfeqwgtl tdcvvmrdpn tkrsrgfgfv 61 tyatveevda amnarphkvd grvvepkrav sredsqrpga hltvkkifvg gikedteehh 121 lrdyfeqygk ievieimtdr gsgkkrgfaf vtfddhdsvd kiviqkyhtv nghncevrka 181 lskqemasas ssqrgrsgsg nfgggrgggf ggndnfgrgg nfsgrggfgg srggggyggs 241 gdgyngfgnd gsnfggggsy ndfgnynnqs snfgpmkggn fggrssgpyg gggqyfakpr 301 nqggyggsss sssygsgrrf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count144 = 0; count144 < 4500; count144++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mcsgsgrrrs slsptmrpgt gaergglmmg hpgmhyapmg mhpmgqranm ppvphgmmpq 61 mmppmggppm gqmpgmmssv mpgmmmshms qasmqpalpp gvnsmdvaag tasgaksmwt 121 ehkspdgrty yyntetkqst wekpddlktp aeqllskcpw keyksdsgkp yyynsqtkes 181 rwakpkeled legyqntiva gslitksnlh amikaeessk qeectttsta pvptteiptt 241 mstmaaaeaa aavvaaaaaa aaaaaaanan astsasntvs gtvpvvpepe vtsivatvvd 301 nentvtiste eqaqltstpa iqdqsvevss ntgeetskqe tvadftpkke eeesqpakkt 361 ytwntkeeak qafkellkek rvpsnasweq amkmiindpr ysalaklsek kqafnaykvq 421 tekeekeear skykeakesf qrflenhekm tsttrykkae qmfgemevwn aiserdrlei 481 yedvlfflsk kekeqakqlr krnwealkni ldnmanvtys ttwseaqqyl mdnptfaede 541 elqnmdkeda licfeehira lekeeeeekq ksllrerrrq rknresfqif ldelhehgql 601 hsmsswmely ptissdirft nmlgqpgsta ldlfkfyved lkaryhdekk iikdilkdkg 661 fvvevnttfe dfvaiisstk rsttldagni klafnsllek aearererek eearkmkrke 721 safksmlkqa appieldavw edirerfvke pafeditles erkrifkdfm hvlehecqhh 781 hsknkkhskk skkhhrkrsr srsgsdsddd dshskkkrqr sesrsasehs ssaesersyk 841 kskkhkkksk krrhksdspe sdaerekdkk ekdresekdr trqrseskhk spkkktgkds 901 gnwdtsgsel segelekrrr tlleqldddq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count145 = 0; count145 < 4500; count145++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matatialqv ngqqgggsep aaaaavvaag dkwkppqgtd sikmengqst aaklglpplt 61 peqqealqka kkyameqsik svlvkqtiah qqqqltnlqm aavtmgfgdp lsplqsmaaq 121 rqralaimcr vyvgsiyyel gedtirqafa pfgpiksidm swdsvtmkhk gfafveyevp 181 eaaqlaleqm nsvmlggrni kvgrpsnigq aqpiidqlae earafnriyv asvhqdlsdd 241 diksvfeafg kiksctlard pttgkhkgyg fieyekaqss qdavssmnlf dlggqylrvg 301 kavtppmpll tpatpgglpp aaavaaaaat akitaqeava gaavlgtlgt pglvspaltl 361 aqplgtlpqa vmaaqapgvi tgvtparppi pvtipsvgvv npilaspptl gllepkkeke 421 eeelfpeser pemlseqehm sisgssarhm vmqkllrkqe stvmvlrnmv dpkdidddle 481 gevteecgkf gavnrviiyq ekqgeeedae iivkifvefs iasethkaiq alngrwfagr 541 kvvaevydqe rfdnsdlsa';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count146 = 0; count146 < 4500; count146++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgsdkrvsrt ersgrygsii drddrderes rsrrrdsdyk rssddrrgdr yddyrdydsp 61 erererrnsd rsedgyhsdg dygehdyrhd isdereskti mlrglpitit esdiremmes 121 fegpqpadvr lmkrktgvsr gfafvefyhl qdatswmean qkklviqgkh iamhysnprp 181 kfedwlcnkc clnnfrkrlk cfrcgadkfd seqevppgtt esvqsvdyyc dtiilrniap 241 htvvdsimta lspyaslavn nirlikdkqt qqnrgfafvq lssamdasql lqilqslhpp 301 lkidgktigv dfaksarkdl vlsdgnrvsa fsvastaiaa aqwsstqsqs geggsvdysy 361 lqpgqdgyaq yaqysqdyqq fyqqqaggle sdassasgta vtttsaavvs qspqlynqts 421 nppgspteea qpstststqa paasptgvvp gtkyavpdts tyqydessgy yydpttglyy 481 dpnsqyyyns ltqqylywdg eketyvpaae ssshqqsglp pakegkekke kpksktaqqi 541 akdmerwaks lnkqkenfkn sfqpvnslre eerresaaad agfalfekkg alaerqqlip 601 elvrngdeen plkrglvaay sgdsdneeel verleseeek ladwkkmacl lcrrqfpnkd 661 alvrhqqlsd lhkqnmdiyr rsrlseqele alelreremk yrdraaerre kygipeppep 721 krkkqfdagt vnyeqptkdg idhsnignkm lqamgwregs glgrkcqgit apieaqvrlk 781 gaglgakgsa yglsgadsyk davrkamfar fteme';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count147 = 0; count147 < 4500; count147++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 meyerrggrg drtgrygatd rsqddggenr srdhdyrdmd yrsypreygs qegkhdydds 61 seeqsaedsy easpgsetqr rrrrrhrhsp tgppgfprdg dyrdqdyrte qgeeeeeeed 121 eeeeekasni vmlrmlpqaa teddirgqlq shgvqarevr lmrnkssgqs rgfafvefsh 181 lqdatrwmea nqhslnilgq kvsmhysdpk pkinedwlcn kcgvqnfkrr ekcfkcgvpk 241 seaeqklplg trldqqtlpl ggrelsqgll plpqpyqaqg vlasqalsqg sepssenand 301 tiilrnlnph stmdsilgal apyavlsssn vrvikdkqtq lnrgfafiql stiveaaqll 361 qilqalhppl tidgktinve fakgskrdma snegsrisaa svastaiaaa qwaisqasqg 421 gegtwatsee ppvdysyyqq degygnsqgt esslyahgyl kgtkgpgitg tkgdptgagp 481 easlepgads vsmqafsraq pgaapgiyqq saeasssqgt aansqsytim spavlkselq 541 spthpssalp patsptaqes ysqypvpdvs tyqydetsgy yydpqtglyy dpnsqyyyna 601 qsqqylywdg errtyvpale qsadghketg apskegkekk ekhktktaqq iakdmerwar 661 slnkqkenfk nsfqpisslr dderresata dagyailekk galaerqhts mdlpklasdd 721 rpspprglva aysgesdsee eqerggpere ekltdwqkla cllcrrqfps kealirhqql 781 sglhkqnlei hrrahlsene lealekndme qmkyrdraae rrekygipep pepkrrkygg 841 istasvdfeq ptrdglgsdn igsrmlqamg wkegsglgrk kqgivtpiea qtrvrgsglg 901 argssygvts tesyketlhk tmvtrfneaq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count148 = 0; count148 < 4500; count148++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslyddlgve tsdsktegws knfkllqsql qvkkaaltqa ksqrtkqstv lapvidlkrg 61 gssddrqivd tpphvaaglk dpvpsgfsag evlipladey dpmfpndyek vvkrqreerq 121 rqrelerqke ieerekrrkd rheasgfarr pdpdsdeded yererrkrsm ggaaiappts 181 lvekdkelpr dfpyeedsrp rsqsskaaip ppvyeeqdrp rsptgpsnsf lanmggtvah 241 kimqkygfre gqglgkheqg lstalsvekt skrggkiivg datekdaskk sdsnplteil 301 kcptkvvllr nmvgagevde dlevetkeec ekygkvgkcv ifeipgapdd eavriflefe 361 rvesaikavv dlngryfggr vvkacfynld kfrvldlaeq v';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count149 = 0; count149 < 4500; count149++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msfpphlnrp pmgipalppg ipppqfpgfp ppvppgtpmi pvpmsimapa ptvlvptvsm 61 vgkhlgarkd hpglkakend encgptttvf vgnisekasd mlirqllakc glvlswkrvq 121 gasgklqafg fceykepest lralrllhdl qigekkllvk vdaktkaqld ewkakkkasn 181 gnarpetvtn ddeealdeet krrdqmikga ievlireyss elnapsqesd shprkkkkek 241 kedifrrfpv aplipyplit kedinaieme edkrdlisre iskfrdthkk leeekgkkek 301 erqeiekerr ererererer errererere rererekeke rerererdrd rdrtkerdrd 361 rdrerdrdrd rerssdrnkd rsrsreksrd rerererere rerererere rerererere 421 rererekdkk rdreedeeda yerrklerkl rekeaayqer lknweirerk ktreyekeae 481 reeerrrema keakrlkefl edydddrddp kyyrgsalqk rlrdrekeme aderdrkrek 541 eeleeirqrl laeghpdpda elqrmeqeae rrrqpqikqe peseeeeeek qekeekreep 601 meeeeepeqk pclkptlrpi ssapsvssas gnatpntpgd espcgiiiph enspdqqqpe 661 ehrpkiglsl klgasnspgq pnsvkrkklp vdsvfnkfed edsddvprkr klvpldyged 721 dknatkgtvn teekrkhiks liekiptakp elfaypldws ivdsilmerr irpwinkkii 781 eyigeeeatl vdfvcskvma hsspqsildd vamvldeeae vfivkmwrll iyeteakkig 841 lvk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count150 = 0; count150 < 4500; count150++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matganatpl dfpskkrkrs rwnqdtmeqk tvipgmptvi ppgltreqer ayivqlqied 61 ltrklrtgdl gippnpedrs pspepiynse gkrlntrefr trkkleeerh nlitemvaln 121 pdfkppadyk ppatrvsdkv mipqdeypei nfvglligpr gntlknieke cnakimirgk 181 gsvkegkvgr kdgqmlpged eplhalvtan tmenvkkave qirnilkqgi etpedqndlr 241 kmqlrelarl ngtlreddnr ilrpwqsset rsitnttvct kcggaghias dckfqrpgdp 301 qsaqdkarmd keylslmael geapvpasvg stsgpattpl asaprpaapa nnppppslms 361 ttqsrppwmn sgpsesrpyh gmhgggpggp gggphsfphp lpsltgghgg hpmqhnpngp 421 pppwmqpppp pmnqgphppg hhgpppmdqy lgstpvgsgv yrlhqgkgmm ppppmgmmpp 481 pppppsgqpp pppsgplppw qqqqqqpppp pppsssmass tplpwqqntt ttttsagtgs 541 ippwqqqqaa aaaspgapqm qgnptmvplp pgvqpplppg appppppppp gsagmmyapp 601 ppppppmdps nfvtmmgmgv agmppfgmpp appppppqn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count151 = 0; count151 < 4500; count151++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msedlakqla sykaqlqqve aalsgngene dllklkkdlq evieltkdll stqpsetlas 61 sdsfastqpt hswkvgdkcm avwsedgqcy eaeieeidee ngtaaitfag ygnaevtpll 121 nlkpveegrk akedsgnkpm skkemiaqqr eykkkkalkk aqrikeleqe redqkvkwqq 181 fnnraysknk kgqvkrsifa spesvtgkvg vgtcgiadkp mtqyqdtsky nvrhlmpq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count152 = 0; count152 < 4500; count152++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslkmdnrdv agkanrwfgv appksgkmnm nilhqeelia qkkreieakm eqkakqnqva 61 spqpphpgei tnahnsscis nkfandgsfl qqflklqkaq tstdaptsap sappstptps 121 agkrsllisr rtglglaslp gpvksyshak qlpvahrpsv fqspdedeee dyeqwleikv 181 sppegaetrk vieklarfva eggpelekva medykdnpaf aflhdknsre flyyrkkvae 241 irkeaqksqa asqkvspped eevknlaekl arfiadggpe vetialqnnr enqafsflye 301 pnsqgykyyr qkleefrkak asstgsftap dpglkrkspp ealsgslppa ttcpasstpa 361 ptiipapaap gkpasaatvk rkrksrwgpe edkvelppae lvqrdvdasp splsvqdlkg 421 lgyekgkpvg lvgvtelsda qkkqlkeqqe mqqmydmimq hkramqdmql lwekavqqhq 481 hgydsdeevd selgtwehql rrmemdktre waeqltkmgr gkhfigdflp pdelekfmet 541 fkalkegrep dyseykefkl tvenigyqml mkmgwkegeg lgsegqgikn pvnkgtttvd 601 gagfgidrpa elskeddeye afrkrmmlay rfrpnplnnp rrpyy';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count153 = 0; count153 < 4500; count153++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msktnksksg srssrsrsas rsrsrsfsks rsrsrslsrs rkrrlssrsr srsyspahnr 61 ernhprvyqn rdfrghnrgy rrpyyfrgrn rgfypwgqyn rggygnyrsn wqnyrqaysp 121 rrgrsrsrsp krrspsprsr shsrnsdkss sdrsrrssss rsssnhsrve sskrksakek 181 kssskdsrps qaagdnqgde akeqtfsggt sqdtkasess kpwpdatygt gsasrasavs 241 elsprerspa lksplqsvvv rrrsprpspv pkpspplsst sqmgstlpsg agyqsgthqg 301 qfdhgsgsls pskkspvgks ppstgstygs sqkeesaasg gaaytkryle eqktengkdk 361 eqkqtntdke kikekgsfsd tglgdgkmks dsfapktdse kpfrgsqspk ryklrddfek 421 kmadfhkeem ddqdkdkakg rkesefddep kfmskvigan knqeeeksgk weglvyappg 481 kekqrkteel eeesfpersk kedrgkrseg ghrgfvpekn frvtaykavq eksssppprk 541 tsesrdklga kgdfptgkss fsitreaqvn vrmdsfdedl arpsgllaqe rklcrdlvhs 601 nkkeqefrsi fqhiqsaqsq rspselfaqh ivtivhhvke hhfgssgmtl herftkylkr 661 gteqeaaknk kspeihrrid ispstfrkhg lahdemkspr epgykaegky kddpvdlrld 721 ierrkkhker dlkrgksres vdsrdsshsr ersaektekt hkgskkqkkh rrardrsrss 781 ssssqsshsy kaeeyteete ereesttgfd ksrlgtkdfv gpsergggra rgtfqfrarg 841 rgwgrgnysg nnnnnsnndf qkrnreeewd peytpkskky ylhddregeg sdkwvsrgrg 901 rgafprgrgr fmfrksstsp kwahdkfsge egeieddesg tenreekdni qptte';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count154 = 0; count154 < 4500; count154++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeylasifg tekdkvncsf yfkigacrhg drcsrlhnkp tfsqtialln iyrnpqnssq 61 sadglrcavs dvemqehyde ffeevfteme ekygeveemn vcdnlgdhlv gnvyvkfrre 121 edaekavidl nnrwfngqpi haelspvtdf reaccrqyem gectrggfcn fmhlkpisre 181 lrrelygrrr kkhrsrsrsr errsrsrdrg rggggggggg gggrerdrrr srdrersgrf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count155 = 0; count155 < 4500; count155++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msdfdeferq lnenkqerdk enrhrkrshs rsrsrdrkrr srsrdrrnrd qrsasrdrrr 61 rskpltrgak eehgglirsp rhekkkkvrk ywdvpppgfe hitpmqykam qaagqipata 121 llptmtpdgl avtptpvpvv gsqmtrqarr lyvgnipfgi teeammdffn aqmrlggltq 181 apgnpvlavq inqdknfafl efrsvdettq amafdgiifq gqslkirrph dyqplpgmse 241 npsvyvpgvv stvvpdsahk lfigglpnyl nddqvkellt sfgplkafnl vkdsatglsk 301 gyafceyvdi nvtdqaiagl ngmqlgdkkl lvqrasvgak natlvsppst inqtpvtlqv 361 pglmssqvqm gghptevlcl mnmvlpeell ddeeyeeive dvrdecskyg lvksieiprp 421 vdgvevpgcg kifveftsvf dcqkamqglt grkfanrvvv tkycdpdsyh rrdfw';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count156 = 0; count156 < 4500; count156++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madktpggsq kassktrssd vhssgssdah mdasgpsdsd mpsrtrpksp rkhnyrnesa 61 reslcdsphq nlsrpllenk lkafsigkms takrtlskke qeelkkkede kaaaeiyeef 121 laafegsdgn kvktfvrggv vnaakeehet dekrgkiykp ssrfadqknp pnqssnerpp 181 sllvietkkp plkkgekekk ksnlelfkee lkqiqeerde rhktkgrlsr feppqsdsdg 241 qrrsmdapsr rnrssgvldd yapgshdvgd psttnlylgn inpqmneeml cqefgrfgpl 301 asvkimwprt deerarernc gfvafmnrrd aeralknlng kmimsfemkl gwgkavpipp 361 hpiyippsmm ehtlppppsg lpfnaqprer lknpnapmlp ppknkedfek tlsqaivkvv 421 ipternllal ihrmiefvvr egpmfeamim nreinnpmfr flfenqtpah vyyrwklysi 481 lqgdsptkwr tedfrmfkng sfwrppplnp ylhgmseeqe teafveepsk kgalkeeqrd 541 kleeilrglt prkndigdam vfclnnaeaa eeivdcites lsilktplpk kiarlylvsd 601 vlynssakva nasyyrkffe tklcqifsdl natyrtiqgh lqsenfkqrv mtcfrawedw 661 aiypepflik lqniflglvn iieeketedv pddldgapie eeldgapled vdgipidatp 721 iddldgvpik sldddldgvp ldatedskkn epifkvapsk weavdesele aqavttskwe 781 lfdqheesee eenqnqeees edeedtqssk seehhlysnp ikeemteskf skysemseek 841 raklreielk vmkfqdeles gkrpkkpgqs fqeqvehyrd kllqrekeke lerererdkk 901 dkeklesrsk dkkekdectp trkerkrrhs tspspsrsss grrvkspspk sersersers 961 hkessrsrss hkdsprdvsk kakrspsgsr tpkrsrrsrs rspkksgkks rsqsrsphrs 1021 hkkskknkh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count157 = 0; count157 < 4500; count157++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maapaqpkki vaptvsqina efvtqlacky waphikkksp fdikviediy ekeivksrfa 61 irkimllefs qylenylwmn yspevsskay lmsiccmvne kfrenvpawe ifkkkpdhfp 121 fffkhilkaa laetdgefsl heqtvlllfl dhcfnslevd lirsqvqqli slpmwmglql 181 arlelelkkt pklrkfwnli kkndekmdpe areqayqerr flsqliqkfi svlksvplse 241 pvtmdkvhyc erfielmidl eallptrrwf ntilddshll vhcylsnlvr reedghlfsq 301 lldmlkfytg feindqtgna ltenemttih ydritslqra afahfpelyd falsnvaevd 361 treslvkffg plssntlhqv asylcllptl pknedttfdk efllellvsr herrisqiqq 421 lnqmplypte kiiwdenivp teyysgegcl alpklnlqfl tlhdyllrnf nlfrlestye 481 irqdiedsvs rmkpwqseyg gvvfggwarm aqpivaftvv evakpnigen wptrvradvt 541 inlnvrdhik deweglrkhd vcflitvrpt kpygtkfdrr rpfieqvglv yvrgceiqgm 601 lddkgrvied gpeprpnlrg esrtfrvfld pnqyqqdmtn tiqngaedvy etfniimrrk 661 pkennfkavl etirnlmntd cvvpdwlhdi ilgygdpssa hyskmpnqia tldfndtfls 721 iehlkasfpg hnvkvtvedp alqippfrit fpvrsgkgkk rkdadveded teeaktlive 781 phvipnrgpy pynqpkrnti qfthtqieai ragmqpgltm vvgppgtgkt dvavqiisni 841 yhnfpeqrtl ivthsnqaln qlfekimald iderhllrlg hgeeeletek dfsrygrvny 901 vlarrielle evkrlqkslg vpgdasytce tagyfflyqv msrweeyisk vknkgstlpd 961 vtevstffpf heyfanapqp ifkgrsyeed meiaegcfrh ikkiftqlee frasellrsg 1021 ldrskyllvk eakiiamtct haalkrhdlv klgfkydnil meeaaqilei etfiplllqn 1081 pqdgfsrlkr wimigdhhql ppviknmafq kysnmeqslf trfvrvgvpt vdldaqgrar 1141 aslcnlynwr yknlgnlphv qllpefstan agllydfqli nvedfqgvge sepnpyfyqn 1201 lgeaeyvval fmymcllgyp adkisiltty ngqkhlirdi inrrcgnnpl igrpnkvttv 1261 drfqgqqndy illslvrtra vghlrdvrrl vvamsrarlg lyifarvslf qncfeltpaf 1321 sqltarplhl hiiptepfpt trkngerpsh evqiiknmpq manfvynmym hliqtthhyh 1381 qtllqlppam veegeevqnq eteleteeea mtvqadiips ptdtscrqet pafqtdttps 1441 etgatstpea ipalsettpt vvgavsapae antpqdatsa peetk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count158 = 0; count158 < 4500; count158++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpkvkrsrka ppdgweliep tldeldqkmr eaetephegk rkveslwpif rihhqktryi 61 fdlfykrkai srelyeycik egyadknlia kwkkqgyenl cclrciqtrd tnfgtncicr 121 vpksklevgr iiecthcgcr gcsg';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count159 = 0; count159 < 4500; count159++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtatvenltf qkdtlgnavd kntsrlelrs yslagrhgst eplvlawssq frrltwgcal 61 dalhrspcva asqhgvthli rssrtphstr crkedaqpgh hgngaasvta qargqrsvlq 121 vplpvprscl fsesfvvsvs sqsrflasvp gtgvqrstaa dmaastaagk qripkvakvk 181 nkapaevqit aeqllreake relellpppp qqkitdeeel ndyklrkrkt fednirknrt 241 visnwikyaq weeslkeiqr arsiyerald vdyrnitlwl kyaememknr qvnharniwd 301 raittlprvn qfwykytyme emlgnvagar qvferwmewq peeqawhsyi nfelrykevd 361 rartiyerfv lvhpdvknwi kyarfeekha yfaharkvye raveffgdeh mdehlyvafa 421 kfeenqkefe rvrviykyal driskqdaqe lfknytifek kfgdrrgied iivskrrfqy 481 eeevkanphn ydawfdylrl vesdaeaeav revyeraian vppiqekrhw kryiylwiny 541 alyeeleakd pertrqvyqa sleliphkkf tfakmwilya qfeirqknls larralgtsi 601 gkcpknklfk vyielelqlr efdrcrklye kflefgpenc tswikfaele tilgdidrar 661 aiyelaisqp rldmpevlwk syidfeieqe etertrnlyr rllqrtqhvk vwisfaqfel 721 ssgkegsltk crqiyeeank tmrnceekee rlmlleswrs feeefgtasd kervdklmpe 781 kvkkrrkvqt ddgsdagwee yfdyifpeda anqpnlklla maklwkkqqq ekedaehhpd 841 edvdeses';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count160 = 0; count160 < 4500; count160++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgresrhyrk rsasrgrsgs rsrsrspsdk rskrgddrrs rsrdrdrrre rsrsrdkrrs 61 rsrdrkrlrr srsrerdrsr errrsrsrdr rrsrsrsrgr rsrssspgnk skktenrsrs 121 kektdggess kekkkdkddk edekekdagn fdqnkleeem rkrkervekw reeqrkkame 181 nigelkkeie emkqgkkwsl edddddeddp aeaekegnem egeeldplda ymeevkeevk 241 kfnmrsvkgg ggnekksgpt vtkvvtvvtt kkavvdsdkk kgelmendqd ameysseeee 301 vdlqtaltgy qtkqrkllep vdhgkieyep frknfyvevp elakmsqeev nvfrlemegi 361 tvkgkgcpkp ikswvqcgis mkilnslkkh gyekptpiqt qaipaimsgr dligiaktgs 421 gktiafllpm frhimdqrsl eegegpiavi mtptrelalq itkeckkfsk tlglrvvcvy 481 ggtgiseqia elkrgaeiiv ctpgrmidml aansgrvtnl rrvtyvvlde adrmfdmgfe 541 pqvmrivdnv rpdrqtvmfs atfprameal arrilskpie vqvggrsvvc sdveqqvivi 601 eeekkflkll ellghyqesg sviifvdkqe hadgllkdlm rasypcmslh ggidqydrds 661 iindfkngtc kllvatsvaa rgldvkhlil vvnyscpnhy edyvhragrt gragnkgyay 721 tfitedqary agdiikalel sgtavppdle klwsdfkdqq kaegkiikks sgfsgkgfkf 781 deteqalane rkklqkaalg lqdsddedaa vdideqiesm fnskkrvkdm aapgtssvpa 841 ptagnaekle iakrlalrin aqknlgiesq vdvmqqatna ilrggtilap tvsaktiaeq 901 laekinakln yvplekqeee rqdggqnesf kryeeelein dfpqtarwkv tskealqris 961 eyseaaitir gtyfppgkep kegerkiyla iesanelavq kakaeitrli keelirlqns 1021 yqptnkgryk vl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count161 = 0; count161 < 4500; count161++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mskrhrldlg edypsgkkra gtdgkdrdrd rdredrskdr drerdrgdre rerekekeke 61 lrastnamli saglpplkas hsahsthsah sthsthsahs thaghaghts lpqcinpftn 121 lphtpryydi lkkrlqlpvw eykdrftdil vrhqsfvlvg etgsgkttqi pqwcveymrs 181 lpgpkrgvac tqprrvaams vaqrvademd vmlgqevgys irfedcssak tilkymtdgm 241 llreamndpl lerygviild eahertlatd ilmgvlkevv rqrsdlkviv msatldagkf 301 qiyfdncpll tipgrthpve ifytpeperd yleaairtvi qihmceeeeg dlllfltgqe 361 eideackrik revddlgpev gdikiiplys tlppqqqqri feppppkkqn gaigrkvvvs 421 tniaetslti dgvvfvidpg fakqkvynpr irvesllvta iskasaqqra gragrtrpgk 481 cfrlytekay ktemqdntyp eilrsnlgsv vlqlkklgid dlvhfdfmdp papetlmral 541 ellnylaaln ddgdltelgs mmaefpldpq lakmviascd yncsnevlsi tamlsvpqcf 601 vrpteakkaa deakmrfahi dgdhltllnv yhafkqnhes vqwcydnfin yrslmsadnv 661 rqqlsrimdr fnlprrstdf tsrdyyinir kalvtgyfmq vahlertghy ltvkdnqvvq 721 lhpstvldhk pewvlynefv lttknyirtc tdikpewlvk iapqyydmsn fpqceakrql 781 driiaklqsk eysqy';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

}

function vital\_components3() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count2 = 0; count2 < 4500; count2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count4 = 0; count4 < 4500; count4++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count6 = 0; count6 < 4500; count6++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSLSNKLTLDKLDVKGKRVVMRVDFNVPMKNNQITNNQRIKAAV PSIKFCLDNGAKSVVLMSHLGRPDGVPMPDKYSLEPVAVELKSLLGKDVLFLKDCVGP EVEKACANPAAGSVILLENLRFHVEEEGKGKDASGNKVKAEPAKIEAFRASLSKLGDV YVNDAFGTAHRAHSSMVGVNLPQKAGGFLMKKELNYFAKALESPERPFLAILGGAKVA DKIQLINNMLDKVNEMIIGGGMAFTFLKVLNNMEIGTSLFDEEGAKIVKDLMSKAEKN GVKITLPVDFVTADKFDENAKTGQATVASGIPAGWMGLDCGPESSKKYAEAVTRAKQI VWNGPVGVFEWEAFARGTKALMDEVVKATSRGCITIIGGGDTATCCAKWNTEDKVSHV STGGGASLELLEGKVLPGVDALSNI';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count8 = 0; count8 < 4500; count8++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MAAYKLVLIRHGESAWNLENRFSGWYDADLSPAGHEEAKRGGQA LRDAGYEFDICFTSVQKRAIRTLWTVLDAIDQMWLPVVRTWRLNERHYGGLTGLNKAE TAAKHGEAQVKIWRRSYDVPPPPMEPDHPFYSNISKDRRYADLTEDQLPSCESLKDTI ARALPFWNEEIVPQIKEGKRVLIAAHGNSLRGIVKHLEGLSEEAIMELNLPTGIPIVY ELDKNLKPIKPMQFLGDEETVRKAMEAVAAQGKAKK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSILKIHAREIFDSRGNPTVEVDLFTSKGLFRAAVPSGASTGIY EALELRDNDKTRYMGKGVSKAVEHINKTIAPALVSKKLNVTEQEKIDKLMIEMDGTEN KSKFGANAILGVSLAVCKAGAVEKGVPLYRHIADLAGNSEVILPVPAFNVINGGSHAG NKLAMQEFMILPVGAANFREAMRIGAEVYHNLKNVIKEKYGKDATNVGDEGGFAPNIL ENKEGLELLKTAIGKAGYTDKVVIGMDVAASEFFRSGKYDLDFKSPDDPSRYISPDQL ADLYKSFIKDYPVVSIEDPFDQDDWGAWQKFTASAGIQVVGDDLTVTNPKRIAKAVNE KSCNCLLLKVNQIGSVTESLQACKLAQANGWGVMVSHRSGETEDTFIADLVVGLCTGQ IKTGAPCRSERLAKYNQLLRIEEELGSKAKFAGRNFRNPLAK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count10 = 0; count10 < 4500; count10++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count12 = 0; count12 < 4500; count12++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count14 = 0; count14 < 4500; count14++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count16 = 0; count16 < 4500; count16++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSLSNKLTLDKLDVKGKRVVMRVDFNVPMKNNQITNNQRIKAAV PSIKFCLDNGAKSVVLMSHLGRPDGVPMPDKYSLEPVAVELKSLLGKDVLFLKDCVGP EVEKACANPAAGSVILLENLRFHVEEEGKGKDASGNKVKAEPAKIEAFRASLSKLGDV YVNDAFGTAHRAHSSMVGVNLPQKAGGFLMKKELNYFAKALESPERPFLAILGGAKVA DKIQLINNMLDKVNEMIIGGGMAFTFLKVLNNMEIGTSLFDEEGAKIVKDLMSKAEKN GVKITLPVDFVTADKFDENAKTGQATVASGIPAGWMGLDCGPESSKKYAEAVTRAKQI VWNGPVGVFEWEAFARGTKALMDEVVKATSRGCITIIGGGDTATCCAKWNTEDKVSHV STGGGASLELLEGKVLPGVDALSNI';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count18 = 0; count18 < 4500; count18++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MAAYKLVLIRHGESAWNLENRFSGWYDADLSPAGHEEAKRGGQA LRDAGYEFDICFTSVQKRAIRTLWTVLDAIDQMWLPVVRTWRLNERHYGGLTGLNKAE TAAKHGEAQVKIWRRSYDVPPPPMEPDHPFYSNISKDRRYADLTEDQLPSCESLKDTI ARALPFWNEEIVPQIKEGKRVLIAAHGNSLRGIVKHLEGLSEEAIMELNLPTGIPIVY ELDKNLKPIKPMQFLGDEETVRKAMEAVAAQGKAKK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSILKIHAREIFDSRGNPTVEVDLFTSKGLFRAAVPSGASTGIY EALELRDNDKTRYMGKGVSKAVEHINKTIAPALVSKKLNVTEQEKIDKLMIEMDGTEN KSKFGANAILGVSLAVCKAGAVEKGVPLYRHIADLAGNSEVILPVPAFNVINGGSHAG NKLAMQEFMILPVGAANFREAMRIGAEVYHNLKNVIKEKYGKDATNVGDEGGFAPNIL ENKEGLELLKTAIGKAGYTDKVVIGMDVAASEFFRSGKYDLDFKSPDDPSRYISPDQL ADLYKSFIKDYPVVSIEDPFDQDDWGAWQKFTASAGIQVVGDDLTVTNPKRIAKAVNE KSCNCLLLKVNQIGSVTESLQACKLAQANGWGVMVSHRSGETEDTFIADLVVGLCTGQ IKTGAPCRSERLAKYNQLLRIEEELGSKAKFAGRNFRNPLAK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count20 = 0; count20 < 4500; count20++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgqicqresr qglamlprlf lnpwrqvill pqppkvlglq ataaekpklh llaeseidky 61 lyamrlsdet lidimtrfrk emknglsrdf nptatvkmlp tfvrsipdgs ekgdfialdl 121 ggssfrilrv qvnheknqnv hmesevydtp enivhgsgsq lfdhvaeclg dfmekrkikd 181 kklpvgftfs fpcqqskide ailitwtkrf kasgvegadv vkllnkaikk rgdydaniva 241 vvndtvgtmm tcgyddqhce vgliigtgtn acymeelrhi dlvegdegrm cintewgafg 301 ddgsledirt efdreidrgs lnpgkqlfek mvsgmylgel vrlilvkmak egllfegrit 361 pelltrgkfn tsdvsaiekn keglhnakei ltrlgvepsd ddcvsvqhvc tivsfrsanl 421 vaatlgailn rlrdnkgtpr lrttvgvdgs lykthpqysr rfhktlrrlv pdsdvrflls 481 esgsgkgaam vtavayrlae qhrqieetla hfhltkdmll evkkrmraem elglrkqthn 541 navvkmlpsf vrrtpdgten gdflaldlgg tnfrvllvki rsgkkrtvem hnkiyaipie 601 imqgtgeelf dhivscisdf ldymgikgpr mplgftfsfp cqqtsldagi litwtkgfka 661 tdcvghdvvt llrdaikrre efdldvvavv ndtvgtmmtc ayeeptcevg livgtgsnac 721 ymeemknvem vegdqgqmci nmewgafgdn gclddirthy drlvdeysln agkqryekmi 781 sgmylgeivr nilidftkkg flfrgqiset lktrgifetk flsqiesdrl allqvrailq 841 qlglnstcdd silvktvcgv vsrraaqlcg agmaavvdki renrgldrln vtvgvdgtly 901 klhphfsrim hqtvkelspk cnvsfllsed gsgkgaalit avgvrlrtea ss';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count22 = 0; count22 < 4500; count22++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvalcslqhl gssdpralpt lptatsgqrp akrrrkspam aaltrdpqfq klqqwyrehr 61 selnlrrlfd ankdrfnhfs ltlntnhghi lvdysknlvt edvmrmlvdl aksrgveaar 121 ermfngekin ytegravlhv alrnrsntpi lvdgkdvmpe vnkvldkmks fcqgplmvte 181 alkpyssggp rvwyvsnidg thiaktlaql npesslfiia sktfttqeti tnaetakewf 241 lqaakdpsav akhfvalstn ttkvkefgid pqnmfefwdw vggryslwsa iglsialhvg 301 fdnfeqllsg ahwmdqhfrt tpleknapvl lallgiwyin cfgcethaml pydqylhrfa 361 ayfqqgdmes ngkyitksgt rvdhqtgpiv wgepgtngqh afyqlihqgt kmipcdflip 421 vqtqhpirkg lhhkillanf laqtealmrg ksteearkel qaagkspedl erllphkvfe 481 gnrptnsivf tkltpfmlga lvamyehkif vqgiiwdins fdqwgvelgk qlakkiepel 541 dgsaqvtshd astnglinfi kqqrearvq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhkdefhlkf fmcviqsrql vrtpqrtage astssmlipk pppktdilks ldtmddpdtv 61 gsipvfktew imtheehhaa ktlgigkaia vltsggdaqg mnaavravvr vgiftgarvf 121 fvhegyqglv dggdhikeat wesvsmmlql ggtvigsarc kdfreregrl raaynlvkrg 181 itnlcviggd gsltgadtfr sewsdllsdl qkagkitdee atkssylniv glvgsidndf 241 cgtdmtigtd salhrimeiv daitttaqsh qrtfvlevmg rhcgylalvt slscgadwvf 301 ipecppdddw eehlcrrlse trtrgsrlni iivaegaidk ngkpitsedi knlvvkrlgy 361 dtrvtvlghv qrggtpsafd rilgsrmgve avmallegtp dtpacvvsls gnqavrlplm 421 ecvqvtkdvt kamdekkfde alklrgrsfm nnwevyklla hvrppvsksg shtvavmnvg 481 apaagmnaav rstvrigliq gnrvlvvhdg feglakgqie eagwsyvggw tgqggsklgt 541 krtlpkksfe qisanitkfn iqglviiggf eaytgglelm egrkqfdelc ipfvvipatv 601 snnvpgsdfs vgadtalnti cttcdrikqs aagtkrrvfi ietmggycgy latmaglaag 661 adaayifeep ftirdlqanv ehlvqkmktt vkrglvlrne kcnenyttdf ifnlyseegk 721 gifdsrknvl ghmqqggspt pfdrnfatkm gakamnwmsg kikesyrngr ifantpdsgc 781 vlgmrkralv fqpvaelkdq tdfehripke qwwlklrpil kilakyeidl dtsdhahleh 841 itrkrsgeaa v';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count24 = 0; count24 < 4500; count24++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpyqypaltp eqkkelsdia hrivapgkgi laadestgsi akrlqsigte nteenrrfyr 61 qllltaddrv npciggvilf hetlyqkadd grpfpqviks kggvvgikvd kgvvplagtn 121 getttqgldg lsercaqykk dgadfakwrc vlkigehtps alaimenanv laryasicqq 181 ngivpivepe ilpdgdhdlk rcqyvtekvl aavykalsdh hiylegtllk pnmvtpghac 241 tqkfsheeia matvtalrrt vppavtgitf lsggqseeea sinlnainkc pllkpwaltf 301 sygralqasa lkawggkken lkaaqeeyvk ralanslacq gkytpsgqag aaaseslfvs 361 nhay';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapsrkffvg gnwkmngrkq slgeligtln aakvpadtev vcapptayid farqkldpki 61 avaaqncykv tngaftgeis pgmikdcgat wvvlghserr hvfgesdeli gqkvahalae 121 glgviacige kldereagit ekvvfeqtkv iadnvkdwsk vvlayepvwa igtgktatpq 181 qaqevheklr gwlksnvsda vaqstriiyg gsvtgatcke lasqpdvdgf lvggaslkpe 241 fvdiinakq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count26 = 0; count26 < 4500; count26++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkvkvgvng fgrigrlvtr aafnsgkvdi vaindpfidl nymvymfqyd sthgkfhgtv 61 kaengklvin gnpitifqer dpskikwgda gaeyvvestg vfttmekaga hlqggakrvi 121 isapsadapm fvmgvnheky dnslkiisna scttnclapl akvihdnfgi veglmttvha 181 itatqktvdg psgklwrdgr galqniipas tgaakavgkv ipelngkltg mafrvptanv 241 svvdltcrle kpakyddikk vvkqasegpl kgilgytehq vvssdfnsdt hsstfdagag 301 ialndhfvkl iswydnefgy snrvvdlmah maske';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count27 = 0; count27 < 4500; count27++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSLSNKLTLDKLDVKGKRVVMRVDFNVPMKNNQITNNQRIKAAV PSIKFCLDNGAKSVVLMSHLGRPDGVPMPDKYSLEPVAVELKSLLGKDVLFLKDCVGP EVEKACANPAAGSVILLENLRFHVEEEGKGKDASGNKVKAEPAKIEAFRASLSKLGDV YVNDAFGTAHRAHSSMVGVNLPQKAGGFLMKKELNYFAKALESPERPFLAILGGAKVA DKIQLINNMLDKVNEMIIGGGMAFTFLKVLNNMEIGTSLFDEEGAKIVKDLMSKAEKN GVKITLPVDFVTADKFDENAKTGQATVASGIPAGWMGLDCGPESSKKYAEAVTRAKQI VWNGPVGVFEWEAFARGTKALMDEVVKATSRGCITIIGGGDTATCCAKWNTEDKVSHV STGGGASLELLEGKVLPGVDALSNI';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count28 = 0; count28 < 4500; count28++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MAAYKLVLIRHGESAWNLENRFSGWYDADLSPAGHEEAKRGGQA LRDAGYEFDICFTSVQKRAIRTLWTVLDAIDQMWLPVVRTWRLNERHYGGLTGLNKAE TAAKHGEAQVKIWRRSYDVPPPPMEPDHPFYSNISKDRRYADLTEDQLPSCESLKDTI ARALPFWNEEIVPQIKEGKRVLIAAHGNSLRGIVKHLEGLSEEAIMELNLPTGIPIVY ELDKNLKPIKPMQFLGDEETVRKAMEAVAAQGKAKK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count29 = 0; count29 < 4500; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSILKIHAREIFDSRGNPTVEVDLFTSKGLFRAAVPSGASTGIY EALELRDNDKTRYMGKGVSKAVEHINKTIAPALVSKKLNVTEQEKIDKLMIEMDGTEN KSKFGANAILGVSLAVCKAGAVEKGVPLYRHIADLAGNSEVILPVPAFNVINGGSHAG NKLAMQEFMILPVGAANFREAMRIGAEVYHNLKNVIKEKYGKDATNVGDEGGFAPNIL ENKEGLELLKTAIGKAGYTDKVVIGMDVAASEFFRSGKYDLDFKSPDDPSRYISPDQL ADLYKSFIKDYPVVSIEDPFDQDDWGAWQKFTASAGIQVVGDDLTVTNPKRIAKAVNE KSCNCLLLKVNQIGSVTESLQACKLAQANGWGVMVSHRSGETEDTFIADLVVGLCTGQ IKTGAPCRSERLAKYNQLLRIEEELGSKAKFAGRNFRNPLAK';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count30 = 0; count30 < 4500; count30++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskphseagt afiqtqqlha amadtflehm crldidsppi tarntgiict igpasrsvet 61 lkemiksgmn varlnfshgt heyhaetikn vrtatesfas dpilyrpvav aldtkgpeir 121 tglikgsgta evelkkgatl kitldnayme kcdenilwld yknickvvev gskiyvddgl 181 islqvkqkga dflvteveng gslgskkgvn lpgaavdlpa vsekdiqdlk fgveqdvdmv 241 fasfirkasd vhevrkvlge kgknikiisk ienhegvrrf deileasdgi mvargdlgie 301 ipaekvflaq kmmigrcnra gkpvicatqm lesmikkprp traegsdvan avldgadcim 361 lsgetakgdy pleavrmqhl iareaeaaiy hlqlfeelrr lapitsdpte atavgaveas 421 fkccsgaiiv ltksgrsahq varyrprapi iavtrnpqta rqahlyrgif pvlckdpvqe 481 awaedvdlrv nfamnvgkar gffkkgdvvi vltgwrpgsg ftntmrvvpv p';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count31 = 0; count31 < 4500; count31++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavgknkrlt kggkkgakkk vvdpfskkdw ydvkapamfn irnigktlvt rtqgtkiasd 61 glkgrvfevs ladlqndeva frkfklited vqgkncltnf hgmdltrdkm csmvkkwqtm 121 ieahvdvktt dgyllrlfcv gftkkrnnqi rktsyaqhqq vrqirkkmme imtrevqtnd 181 lkevvnklip dsigkdieka cqsiyplhdv fvrkvkmlkk pkfelgklme lhgegsssgk 241 atgdetgakv eradgyeppv qesv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count32 = 0; count32 < 4500; count32++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msgaldvlqm keedvlkfla agthlggtnl dfqmeqyiyk rksdgiyiin lkrtweklll 61 aaraivaien padvsvissr ntgqravlkf aaatgatpia grftpgtftn qiqaafrepr 121 llvvtdprad hqplteasyv nlptialcnt dsplryvdia ipcnnkgahs vglmwwmlar 181 evlrmrgtis rehpwevmpd lyfyrdpeei ekeeqaaaek avtkeefqge wtapapefta 241 tqpevadwse gvqvpsvpiq qfptedwsaq patedwsaap taqatewvga ttdws';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count33 = 0; count33 < 4500; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavqiskkrk fvadgifkae lnefltrela edgysgvevr vtptrteiii latrtqnvlg 61 ekgrrirelt avvqkrfgfp egsvelyaek vatrglcaia qaeslrykll gglavrracy 121 gvlrfimesg akgcevvvsg klrgqraksm kfvdglmihs gdpvnyyvdt avrhvllrqg 181 vlgikvkiml pwdptgkigp kkplpdhvsi vepkdeilpt tpiseqkggk peppampqpv 241 pta';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count34 = 0; count34 < 4500; count34++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpvarswvcr ktyvtprrpf eksrldqelk ligeyglrnk revwrvkftl akirkaarel 61 ltldekdprr lfegnallrr lvrigvldeg kmkldyilgl kiedflerrl qtqvfklgla 121 ksihharvli rqrhirvrkq vvnipsfivr ldsqkhidfs lrspygggrp grvkrknakk 181 gqggagagdd eeed';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count35 = 0; count35 < 4500; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 margpkkhlk rvaapkhwml dkltgvfapr pstgphklre clpliiflrn rlkyaltgde 61 vkkicmqrfi kidgkvrtdi typagfmdvi sidktgenfr liydtkgrfa vhritpeeak 121 yklckvrkif vgtkgiphlv thdartiryp dplikvndti qidletgkit dfikfdtgnl 181 cmvtgganlg rigvitnrer hpgsfdvvhv kdangnsfat rlsnifvigk gnkpwislpr 241 gkgirltiae erdkrlaakq ssg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count36 = 0; count36 < 4500; count36++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maddagaagg pggpggpgmg nrggfrggfg sgirgrgrgr grgrgrgrga rggkaedkew 61 mpvtklgrlv kdmkikslee iylfslpike seiidfflga slkdevlkim pvqkqtragq 121 rtrfkafvai gdynghvglg vkcskevata irgaiilakl sivpvrrgyw gnkigkphtv 181 pckvtgrcgs vlvrlipapr gtgivsapvp kkllmmagid dcytsargct atlgnfakat 241 fdaisktysy ltpdlwketv ftkspyqeft dhlvkthtrv svqrtqapav att';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mklnisfpat gcqklievdd erklrtfyek rmatevaada lgeewkgyvv risggndkqg 61 fpmkqgvlth grvrlllskg hscyrprrtg erkrksvrgc ivdanlsvln lvivkkgekd 121 ipgltdttvp rrlgpkrasr irklfnlske ddvrqyvvrk plnkegkkpr tkapkiqrlv 181 tprvlqhkrr rialkkqrtk knkeeaaeya kllakrmkea kekrqeqiak rrrlsslras 241 tsksessqk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count38 = 0; count38 < 4500; count38++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtewetaapa vaetpdiklf gkwstddvqi ndislqdyia vkekyakylp hsagryaakr 61 frkaqcpive rltnsmmmhg rnngkklmtv rivkhafeii hlltgenplq vlvnaiinsg 121 predstrigr agtvrrqavd vsplrrvnqa iwllctgare aafrniktia ecladelina 181 akgssnsyai kkkdelerva ksnr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfsssakivk pngekpdefe sgisqallel emnsdlkaql relnitaake ievgggrkai 61 iifvpvpqlk sfqkiqvrlv relekkfsgk hvvfiaqrri lpkptrksrt knkqkrprsr 121 tltavhdail edlvfpseiv gkrirvkldg srlikvhldk aqqnnvehkv etfsgvykkl 181 tgkdvnfefp efql';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count40 = 0; count40 < 4500; count40++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvrmnvlada lksinnaekr gkrqvlirpc skvivrfltv mmkhgyigef eiiddhragk 61 ivvnltgrln kcgvisprfd vqlkdlekwq nnllpsrqfg fivlttsagi mdheearrkh 121 tggkilgfff';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgisrdnwhk rrktggkrkp yhkkrkyelg rpaantkigp rrihtvrvrg gnkkyralrl 61 dvgnfswgse cctrktriid vvynasnnel vrtktlvknc ivlidstpyr qwyeshyalp 121 lgrkkgaklt peeeeilnkk rskkiqkkyd erkknakiss lleeqfqqgk llaciasrpg 181 qcgradgyvl egkelefylr kikarkgk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count42 = 0; count42 < 4500; count42++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpskgplqsv qvfgrkktat avahckrgng likvngrple mieprtlqyk llepvlllgk 61 erfagvdirv rvkggghvaq iyairqsisk alvayyqkyv deaskkeikd iliqydrtll 121 vadprrcesk kfggpgarar yqksyr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mafkdtgktp vepevaihri ritltsrnvk slekvcadli rgakeknlkv kgpvrmptkt 61 lrittrktpc gegsktwdrf qmrihkrlid lhspseivkq itsisiepgv eliestdaep 121 mdtegqqytl rsvfespgtc pf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count44 = 0; count44 < 4500; count44++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlmpkknria iyellfkegv mvakkdvhmp khpeladknv pnlhvmkamq slksrgyvke 61 qfawrhfywy ltnegiqylr dylhlppeiv patlrrsrpe tgrprpkgle gerparltrg 121 eadrdtyrrs avppgadkka eagagsatef qfrggfgrgr gqppq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maprkgkekk eeqvislgpq vaegenvfgv chifasfndt fvhvtdlsgk eticrvtggm 61 kvkadrdess pyaamlaaqd vaqrckelgi talhiklrat ggnrtktpgp gaqsalrala 121 rsgmkigrie dvtpipsdst rrkggrrgrr l';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count46 = 0; count46 < 4500; count46++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkcrglrta rklrshrrdq kwhdkqykka hlgtalkanp fggashakgi vlekvgveak 61 qpnsairkcv rvqlikngkk itafvpndgc lnfieendev lvagfgrkgh avgdipgvrf 121 kvvkvanvsl lalykgkker prs';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeegiaagg vmdvntalqe vlktalihdg largireaak aldkrqahlc vlasncdepm 61 yvklvealca ehqinlikvd dnkklgewvg lckidregkp rkvvgcscvv vkdygkesqa 121 kdvieeyfkc kk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count48 = 0; count48 < 4500; count48++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mslvipekfq hilrvlntni dgrrkiafai taikgvgrry ahvvlrkadi dltkragelt 61 edeverviti mqnprqykip dwflnrqkdv kdgkysqvla ngldnklred lerlkkirah 121 rglrhfwglr vrgqhtkttg rrgrtvgvsk kk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mghqqlywsh prkfgqgsrs crvcsnrhgl irkyglnmcr qcfrqyakdi gfikld';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count50 = 0; count50 < 4500; count50++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrmhapgkg lsqsalpyrr svptwlklts ddvkeqiykl akkgltpsqi gvilrdshgv 61 aqvrfvtgnk ilrilkskgl apdlpedlyh likkavavrk hlernrkdkd akfrlilies 121 rihrlaryyk tkrvlppnwk yesstasalv a';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madiqteray qkqptifqnk krvllgetgk eklpryykni glgfktpkea iegtyidkkc 61 pftgnvsirg rilsgvvtkm kmqrtivirr dylhyirkyn rfekrhknms vhlspcfrdv 121 qigdivtvge crplsktvrf nvlkvtkaag tkkqfqkf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count52 = 0; count52 < 4500; count52++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrvrtktvk kaarviieky ytrlgndfht nkrvceeiai ipskklrnki agyvthlmkr 61 iqrgpvrgis iklqeeerer rdnyvpevsa ldqeiievdp dtkemlklld fgslsnlqvt 121 qptvgmnfkt prgpv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlgrgadlae veqkkkrtfr kftyrgvdld qlldmsyeql mqlysarqrr rlnrglrrkq 61 hsllkrlrka kkeappmekp evvkthlrdm iilpemvgsm vgvyngktfn qveikpemig 121 hylgefsity kpvkhgrpgi gathssrfip lk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count54 = 0; count54 < 4500; count54++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpgvtvkdvn qqefvralaa flkksgklkv pewvdtvkla khkelapyde nwfytraast 61 arhlylrgga gvgsmtkiyg grqrngvmps hfsrgsksva rrvlqalegl kmvekdqdgg 121 rkltpqgqrd ldriagqvaa ankkh';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqndagefvd lyvprkcsas nriigakdha siqmnvaevd kvtgrfngqf ktyaicgair 61 rmgesddsil rlakadgivs knf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count56 = 0; count56 < 4500; count56++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mndtvtirtr kfmtnrllqr kqmvidvlhp gkatvpktei reklakmykt tpdvifvfgf 61 rthfgggktt gfgmiydsld yakknepkhr larhglyekk ktsrkqrker knrmkkvrgt 121 akanvgagkk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mppkddkkkk dagksakkdk dpvnksggka kkkkwskgkv rdklnnlvlf dkatydklck 61 evpnyklitp avvserlkir gslaraalqe llskgliklv skhraqviyt rntkggdapa 121 ageda';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count58 = 0; count58 < 4500; count58++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtkkrrnngr akkgrghvqp irctncarcv pkdkaikkfv irniveaaav rdiseasvfd 61 ayvlpklyvk lhycvscaih skvvrnrsre arkdrtpppr frpagaaprp ppkpm';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mplakdllhp speeekrkhk kkrlvqspns yfmdvkcpgc ykittvfsha qtvvlcvgcs 61 tvlcqptggk arltegcsfr rkqh';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count60 = 0; count60 < 4500; count60++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdtsrvqpik larvtkvlgr tgsqgqctqv rvefmddtsr siirnvkgpv regdvltlle 61 serearrlr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madiqteray qkqptifqnk krvllgetgk eklpryykni glgfktpkea iegtyidkkc 61 pftgnvsirg rilsgvvtkm kmqrtivirr dylhyirkyn rfekrhknms vhlspcfrdv 121 qigdivtvge crplsktvrf nvlkvtkaag tkkqfqkf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count62 = 0; count62 < 4500; count62++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqlfvraqel htfevtgqet vaqikahvas legiapedqv vllagapled eatlgqcgve 61 alttlevagr mlggkvhgsl aragkvrgqt pkvakqekkk kktgrakrrm qynrrfvnvv 121 ptfgkkkgpn ans';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqifvktltg ktitleveps dtienvkaki qdkegippdq qrlifagkql edgrtlsdyn 61 iqkvclgeeq pllkknvile ifyftffetg sdsvt';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count64 = 0; count64 < 4500; count64++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqifvktltg ktitleveps dtienvkaki qdkegippdq qrlifagkql edgrtlsdyn 61 iqkestlhlv lrlrggakkr kkksyttpkk nkhkrkkvkl avlkyykvde ngkisrlrre 121 cpsdecgagv fmashfdrhy cgkccltycf nkpedk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mteqmtlrgt lkghngwvtq iattpqfpdm ilsasrdkti imwkltrdet nygipqralr 61 ghshfvsdvv issdgqfals gswdgtlrlw dlttgtttrr fvghtkdvls vafssdnrqi 121 vsgsrdktik lwntlgvcky tvqdeshsew vscvrfspns snpiivscgw dklvkvwnla 181 ncklktnhig htgylntvtv spdgslcasg gkdgqamlwd lnegkhlytl dggdiinalc 241 fspnrywlca atgpsikiwd legkiivdel kqevistssk aeppqctsla wsadgqtlfa 301 gytdnlvrvw qvtigtr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count66 = 0; count66 < 4500; count66++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msskvsrdtl yeavrevlhg nqrkrrkfle tvelqislkn ydpqkdkrfs gtvrlkstpr 61 pkfsvcvlgd qqhcdeakav diphmdieal kklnknkklv kklakkydaf laseslikqi 121 prilgpglnk agkfpsllth nenmvakvde vkstikfqmk kvlclavavg hvkmtddelv 181 ynihlavnfl vsllkknwqn vralyikstm gkpqrly';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrvirgqrk gagsvfrahv khrkgaarlr avdfaerhgy ikgivkdiih dpgrgaplak 61 vvfrdpyrfk krtelfiaae gihtgqfvyc gkkaqlnign vlpvgtmpeg tivccleekp 121 gdrgklaras gnyatvishn petkktrvkl psgskkviss anravvgvva gggridkpil 181 kagrayhkyk akrncwprvr gvamnpvehp fgggnhqhig kpstirrdap agrkvgliaa 241 rrtgrlrgtk tvqeken';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count68 = 0; count68 < 4500; count68++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mshrkfsapr hgslgflprk rssrhrgkvk sfpkddpskp vhltaflgyk agmthivrev 61 drpgskvnkk evveavtive tppmvvvgiv gyvetprglr tfktvfaehi sdeckrrfyk 121 nwhkskkkaf tkyckkwqde dgkkqlekdf ssmkkycqvi rviahtqmrl lplrqkkahl 181 meiqvnggtv aekldwarer leqqvpvnqv fgqdemidvi gvtkgkgykg vtsrwhtkkl 241 prkthrglrk vacigawhpa rvafsvarag qkgyhhrtei nkkiykigqg ylikdgklik 301 nnastdydls dksinplggf vhygevtndf vmlkgcvvgt kkrvltlrks llvqtkrral 361 ekidlkfidt tskfghgrfq tmeekkafmg plkkdriake ega';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 macarplisv ysekgessgk nvtlpavfka pirpdivnfv htnlrknnrq pyavselagh 61 qtsaeswgtg ravariprvr gggthrsgqg afgnmcrggr mfaptktwrr whrrvnttqk 121 ryaicsalaa salpalvmsk ghrieevpel plvvedkveg ykktkeavll lkklkawndi 181 kkvyasqrmr agkgkmrnrr riqrrgpcii ynedngiika frnipgitll nvsklnilkl 241 apgghvgrfc iwtesafrkl delygtwrka aslksnynlp mhkmintdls rilkspeiqr 301 alraprkkih rrvlkknplk nlrimlklnp yaktmrrnti lrqarnhklr vdkaaaaaaa 361 lqaksdekaa vagkkpvvgk kgkkaavgvk kqkkplvgkk aaatkkpape kkpaekkptt 421 eekkpaa';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count70 = 0; count70 < 4500; count70++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maqdqgeken pmrelrirkl clnicvgesg drltraakvl eqltgqtpvf skarytvrsf 61 girrnekiav hctvrgakae eilekglkvr eyelrknnfs dtgnfgfgiq ehidlgikyd 121 psigiygldf yvvlgrpgfs iadkkrrtgc igakhriske eamrwfqqky dgiilpgk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madiqteray qkqptifqnk krvllgetgk eklpryykni glgfktpkea iegtyidkkc 61 pftgnvsirg rilsgvvtkm kmqrtivirr dylhyirkyn rfekrhknms vhlspcfrdv 121 qigdivtvge crplsktvrf nvlkvtkaag tkkqfqkf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count72 = 0; count72 < 4500; count72++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mktilsnqtv dipenvditl kgrtvivkgp rgtlrrdfnh invelsllgk kkkrlrvdkw 61 wgnrkelatv rticshvqnm ikgvtlgfry kmrsvyahfp invviqengs lveirnflge 121 kyirrvrmrp gvacsvsqaq kdelilegnd ielvsnsaal iqqattvknk dirkfldgiy 181 vsekgtvqqa de';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mktilsnqtv dipenvditl kgrtvivkgp rgtlrrdfnh invelsllgk kkkrlrvdkw 61 wgnrkelatv rticshvqnm ikgvtlgfry kmrsvyahfp invviqengs lveirnflge 121 kyirrvrmrp gvacsvsqaq kdelilegnd ielvsnsaal iqqattvknk dirkfldgiy 181 vsekgtvqqa de';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count74 = 0; count74 < 4500; count74++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 magekvekpd tkekkpeakk vdaggkvkkg nlkakkpkkg kphcsrnpvl vrgigrysrs 61 amysrkamyk rkysaakskv ekkkkekvla tvtkpvggdk nggtrvvklr kmpryypted 121 vprkllshgk kpfsqhvrkl rasitpgtil iiltgrhrgk rvvflkqlas glllvtgplv 181 lnrvplrrth qkfviatstk idisnvkipk hltdayfkkk klrkprhqeg eifdtekeky 241 eiteqrkidq kavdsqilpk ikaipqlqgy lrsvfaltng iyphklvf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpkgkkakgk kvapapavvk kqeakkvvnp lfekrpknfg igqdiqpkrd ltrfvkwpry 61 irlqrqrail ykrlkvppai nqftqaldrq tatqllklah kyrpetkqek kqrllaraek 121 kaagkgdvpt krppvlragv ntvttlvenk kaqlvviahd vdpielvvfl palcrkmgvp 181 yciikgkarl grlvhrktct tvaftqvnse dkgalaklve airtnyndry deirrhwggn 241 vlgpksvari aklekakake latklg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count76 = 0; count76 < 4500; count76++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpredratwk snyflkiiql lddypkcfiv gadnvgskqm qqirmslrgk avvlmgkntm 61 mrkairghle nnpalekllp hirgnvgfvf tkedlteird mllankvpaa aragaiapce 121 vtvpaqntgl gpektsffqa lgittkisrg tieilsdvql iktgdkvgas eatllnmlni 181 spfsfglviq qvfdngsiyn pevlditeet lhsrflegvr nvasvclqig yptvasvphs 241 iingykrvla lsvetdytfp laekvkafla dpsafvaaap vaaattaapa aaaapakvea 301 keeseesded mgfglfd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mppkfdpnei kvvylrctgg evgatsalap kigplglspk kvgddiakat gdwkglritv 61 kltiqnrqaq ievvpsasal iikalkeppr drkkqknikh sgnitfdeiv niarqmrhrs 121 larelsgtik eilgtaqsvg cnvdgrhphd iiddinsgav ecpas';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count78 = 0; count78 < 4500; count78++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maevqvlvld grghllgrla aivakqvllg rkvvvvrceg inisgnfyrn klkylaflrk 61 rmntnpsrgp yhfrapsrif wrtvrgmlph ktkrgqaald rlkvfdgipp pydkkkrmvv 121 paalkvvrlk ptrkfaylgr lahevgwkyq avtatleekr kekakihyrk kkqlmrlrkq 181 aeknvekkid kytevlkthg llv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapsrngmvl kphfhkdwqr rvatwfnqpa rkirrrkarq akarriaprp asgpirpivr 61 cptvryhtkv ragrgfslee lrvagihkkv artigisvdp rrrnkstesl qanvqrlkey 121 rsklilfprk psapkkgdss aeelklatql tgpvmpvrnv ykkekarvit eeeknfkafa 181 slrmaranar lfgirakrak eaaeqdvekk k';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count80 = 0; count80 < 4500; count80++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mskrgrggss gakfrislgl pvgavincad ntgaknlyii svkgikgrln rlpaagvgdm 61 vmatvkkgkp elrkkvhpav virqrksyrr kdgvflyfed nagvivnnkg emkgsaitgp 121 vakecadlwp riasnagsia';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvfrrfvevg rvayvsfgph agklvaivdv idqnralvdg pctqvrrqam pfkcmqltdf 61 ilkfphsahq kyvrqawqka dintkwaatr wakkiearer kakmtdfdrf kvmkakkmrn 121 riiknevkkl qkaallkasp kkapgtkgta aaaaaaaaak vpakkitaas kkapaqkvpa 181 qkatgqkaap apkaqkgqka paqkapapka sgkka';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count82 = 0; count82 < 4500; count82++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpsrlrktrk lrghvshghg rigkhrkhpg grgnagglhh hrinfdkyhp gyfgkvgmkh 61 yhlkrnqsfc ptvnldklwt lvseqtrvna aknktgaapi idvvrsgyyk vlgkgklpkq 121 pvivkakffs rraeekiksv ggacvlva';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgaykyiqel wrkkqsdvmr fllrvrcwqy rqlsalhrap rptrpdkarr lgykakqgyv 61 iyrirvrrgg rkrpvpkgat ygkpvhhgvn qlkfarslqs vaeeragrhc galrvlnsyw 121 vgedstykff evilidpfhk airrnpdtqw itkpvhkhre mrgltsagrk srglgkghkf 181 hhtiggsrra awrrrntlql hryr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count84 = 0; count84 < 4500; count84++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrrparcyr ycknkpypks rfcrgvpdak irifdlgrkk akvdefplcg hmvsdeyeql 61 ssealeaari cankymvksc gkdgfhirvr lhpfhvirin kmlscagadr lqtgmrgafg 121 kpqgtvarvh igqvimsirt klqnkehvie alrrakfkfp grqkihiskk wgftkfnade 181 fedmvaekrl ipdgcgvkyi psrgpldkwr alhs';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgfvkvvknk ayfkryqvkf rrrregktdy yarkrlviqd knkyntpkyr mivrvtnrdi 61 icqiayarie gdmivcaaya helpkygvkv gltnyaaayc tglllarrll nrfgmdkiye 121 gqvevtgdey nvesidgqpg aftcyldagl artttgnkvf galkgavdgg lsiphstkrf 181 pgydseskef naevhrkhim gqnvadymry lmeededayk kqfsqyikns vtpdmmeemy 241 kkahaairen pvyekkpkke vkkkrwnrpk mslaqkkdrv aqkkasflra qeraaes';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count86 = 0; count86 < 4500; count86++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgvdirhnkd rkvrrkepks qdiylrllvk lyrflarrtn stfnqvvlkr lfmsrtnrpp 61 lslsrmirkm klpgrenkta vvvgtitddv rvqevpklkv calrvtsrar srilraggki 121 ltfdqlalds pkgcgtvlls gprkgrevyr hfgkapgtph shtkpyvrsk grkferargr 181 rasrgykn';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msmlrlqkrl assvlrcgkk kvwldpnetn eianansrqq irklikdgli irkpvtvhsr 61 arcrkntlar rkgrhmgigk rkgtanarmp ekvtwmrrmr ilrrllrryr eskkidrhmy 121 hslylkvkgn vfknkrilme hihklkadka rkklladqae arrsktkear krreerlqak 181 keeiiktlsk eeetkk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count88 = 0; count88 < 4500; count88++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mkasgtlrey kvvgrclptp kchtpplyrm rifapnhvva ksrfwyfvsq lkkmkkssge 61 ivycgqvfek splrvknfgi wlrydsrsgt hnmyreyrdl ttagavtqcy rdmgarhrar 121 ahsiqimkve eiaaskcrrp avkqfhdski kfplphrvlr rqhkprfttk rpntff';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtntkgkrrg trymfsrpfr khgvvplaty mriykkgdiv dikgmgtvqk gmphkcyhgk 61 tgrvynvtqh avgivvnkqv kgkilakrin vriehikhsk srdsflkrvk endqkkkeak 121 ekgtwvqlkr qpappreahf vrtngkepel lepipyefma';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count90 = 0; count90 < 4500; count90++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvrysldpen ptkscksrgs nlrvhfkntr etaqaikgmh irkatkylkd vtlqkqcvpf 61 rrynggvgrc aqakqwgwtq grwpkksaef llhmlknaes naelkgldvd slviehiqvn 121 kapkmrrrty rahgrinpym sspchiemil tekeqivpkp eeevaqkkki sqkklkkqkl 181 mare';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapvkklvvk ggkkkkqvlk ftldcthpve dgimdaanfe qflqerikvn gkagnlgggv 61 vtierskski tvtsevpfsk rylkyltkky lkknnlrdwl rvvanskesy elryfqinqd 121 eeeeeded';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count92 = 0; count92 < 4500; count92++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapkakkeap appkaeakak alkakkavlk gvhshkkkki rtsptfrrpk tlrlrrqpky 61 prksaprrnk ldhyaiikfp lttesamkki ednntlvfiv dvkankhqik qavkklydid 121 vakvntlirp dgekkayvrl apdydaldva nkigii';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkfnpfvtsd rsknrkrhfn apshirrkim ssplskelrq kynvrsmpir kddevqvvrg 61 hykgqqigkv vqvyrkkyvi yiervqreka ngttvhvgih pskvvitrlk ldkdrkkile 121 rkaksrqvgk ekgkykeeti ekmqe';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count94 = 0; count94 < 4500; count94++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkvelcsfsg ykiypghgrr yartdgkvfq flnakcesaf lskrnprqin wtvlyrrkhk 61 kgqseeiqkk rtrravkfqr aitgasladi makrnqkpev rkaqreqair aakeakkakq 121 askktamaaa kaptkaapkq kivkpvkvsa prvggkr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkfmkpgkv vlvlagrysg rkavivknid dgtsdrpysh alvagidryp rkvtaamgkk 61 kiakrskiks fvkvynynhl mptrysvdip ldktvvnkdv frdpalkrka rreakvkfee 121 ryktgknkwf fqklrf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count96 = 0; count96 < 4500; count96++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msahlqwmvv rncssflikr nkqtystepn nlkarnsfry nglihrktvg vepaadgkgv 61 vvvikrrsgq rkpatsyvrt tinknaratl ssirhmirkn kyrpdlrmvs wglgirlget 121 gqccgegppt tgcnmgwrgm dscfqptpht qhwprgrlve cmg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 makikardlr gkkkeellkq lddlkvelsq lrvakvtgga asklskirvv rksiarvltv 61 inqtqkenlr kfykgkkykp ldlrpkktra mrrrlnkhee nlktkkqqrk erlyplrkya 121 vka';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count98 = 0; count98 < 4500; count98++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maksknhtth nqsrkwhrng ikkprsqrye slkgvdpkfl rnmrfakkhn kkglkkmqan 61 nakamsarae aikalvkpke vkpkipkgvs rkldrlayia hpklgkrara riakglrlcr 121 pkakakakak dqtkaqaaap asvpaqapkr tqaptkase';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 megveekkke vpavpetlkk krrnfaelki krlrkkfaqk mlrkarrkli yekakhyhke 61 yrqmyrteir marmarkagn fyvpaepkla fvirirging vspkvrkvlq llrlrqifng 121 tfvklnkasi nmlrivepyi awgypnlksv neliykrgyg kinkkrialt dnaliarslg 181 kygiicmedl iheiytvgkr fkeannflwp fklssprggm kkktthfveg gdagnredqi 241 nrlirrmn';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count100 = 0; count100 < 4500; count100++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mvaakktkks lesinsrlql vmksgkyvlg ykqtlkmirq gkaklvilan ncpalrksei 61 eyyamlaktg vhhysgnnie lgtacgkyyr vctlaiidpg dsdiirsmpe qtgek';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mapakkggek kkgrsainev vtreytinih krihgvgfkk rapralkeir kfamkemgtp 61 dvridtrlnk avwakgirnv pyrirvrlsr krnededspn klytlvtyvp vttfknlqtv 121 nvden';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count102 = 0; count102 < 4500; count102++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maalrplvkp kivkkrtkkf irhqsdryvk ikrnwrkprg idnrvrrrfk gqilmpnigy 61 gsnkktkhml psgfrkflvh nvkelevllm cnksycaeia hnvssknrka iveraaqlai 121 rvtnpnarlr seene';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msgrlwskai fagykrglrn qrehtallki egvyardete fylgkrcayv ykaknntvtp 61 ggkpnktrvi wgkvtrahgn sgmvrakfrs nlpakaighr irvmlypsri';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count104 = 0; count104 < 4500; count104++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvqrltyrrr lsyntasnkt rlsrtpgnri vylytkkvgk apksacgvcp grlrgvravr 61 pkvlmrlskt kkhvsraygg smcakcvrdr ikraflieeq kivvkvlkaq aqsqkak';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 malrypmavg lnkghkvtkn vskprhsrrr grltkhtkfv rdmirevcgf apyerramel 61 lkvskdkral kfikkrvgth irakrkreel snvlaamrka aakkd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count106 = 0; count106 < 4500; count106++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtkgtssfgk rrnkthtlcr rcgskayhlq kstcgkcgyp akrkrkynws akakrrnttg 61 tgrmrhlkiv yrrfrhgfre gttpkpkraa vaassss';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mprkieeikd flltarrkda ksvkikknkd nvkfkvrcsr ylytlvitdk ekaeklkqsl 61 ppglavkelk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count108 = 0; count108 < 4500; count108++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msshktfrik rflakkqkqn rpipqwirmk tgnkirynsk rrhwrrtklg l';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count109 = 0; count109 < 4500; count109++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqifvktltg ktitleveps dtienvkaki qdkegippdq qrlifagkql edgrtlsdyn 61 iqkestlhlv lrlrggiiep slrqlaqkyn cdkmicrkcy arlhpravnc rkkkcghtnn 121 lrpkkkvk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count110 = 0; count110 < 4500; count110++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'iiep slrqlaqkyn cdkmicrkcy arlhpravnc rkkkcghtnn 121 lrpkkkvk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count111 = 0; count111 < 4500; count111++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'akkr kkksyttpkk nkhkrkkvkl avlkyykvde ngkisrlrre 121 cpsdecgagv fmashfdrhy cgkccltycf nkpedk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count112 = 0; count112 < 4500; count112++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrakwrkkrm rrlkrkrrkm rqrsk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count113 = 0; count113 < 4500; count113++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvnvpktrrt fckkcgkhqp hkvtqykkgk dslyaqgkrr ydrkqsgygg qtkpifrkka 61 kttkkivlrl ecvepncrsk rmlaikrckh felggdkkrk gqviqf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count114 = 0; count114 < 4500; count114++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 makrtkkvgi vgkygtryga slrkmvkkie isqhakytcs fcgktkmkrr avgiwhcgsc 61 mktvaggawt ynttsavtvk sairrlkelk dq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count115 = 0; count115 < 4500; count115++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masvselaci ysalilhdde vtvtedkina likaagvnve pfwpglfaka lanvnigsli 61 cnvgaggpap aagaapaggp apstaaapae ekkveakkee seesdddmgf glfd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count116 = 0; count116 < 4500; count116++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mryvasylla alggnsspsa kdikkildsv gieadddrln kviselngkn iedviaqgig 61 klasvpagga vavsaapgsa apaagsapaa aeekkdekke eseesdddmg fglfd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count117 = 0; count117 < 4500; count117++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = 'G U C U A C G G C C A U A C C A C C C U G A A C G C G C C C G A U C U C G U C U G A U C U C G G A A G C U A A G C A G G G U C G G G C C U G G U U A G U A C U U G G A U G G G A G A C C G C C U G G G A A U A C C G G G U G C U G U A G G C U U U';

rna();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count118 = 0; count118 < 4500; count118++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = 'C G A C U C U U A G C G G U G G A U C A C U C G G C U C G U G C G U C G A U G A A G A A C G C A G C U A G C U G C G A G A A U U A A U G U G A A U U G C A G G A C A C A U U G A U C A U C G A C A C U U C G A A C G C A C U U G C G G C C C C G G G U U C C U C C C G G G G C U A C G C C U G U C U G A G C G U C G C U U ';

rna();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count119 = 0; count119 < 4500; count119++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

rnaseq = 'CGCGACCUCAGAUCAGACGUGGCGACCCGCUGAAUUUAAGCAUAUUAGUCAGCGGAGGAGAAGAAACUAACCAGGAUUCCCUCAGUAACGGCGAGUGAACAGGGAAGAGCCCAGCGCCGAAUCCCCGCCCCGCGGCGGGGCGCGGGACAUGUGGCGUACGGAAGACCCGCUCCCCGGCGCCGCUCGUGGGGGGCCCAAGUCCUUCUGAUCGAGGCCCAGCCCGUGGACGGUGUGAGGCCGGUAGCGGCCCCCGGCGCGCCGGGCCCGGGUCUUCCCGGAGUCGGGUUGCUUGGGAAUGCAGCCCAAAGCGGGUGGUAAACUCCAUCUAAGGCUAAAUACCGGCACGAGACCGAUAGUCAACAAGUACCGUAAGGGAAAGUUGAAAAGAACUUUGAAGAGAGAGUUCAAGAGGGCGUGAAACCGUUAAGAGGUAAACGGGUGGGGUCCGCGCAGUCCGCCCGGAGGAUUCAACCCGGCGGCGGGUCCGGCCGUGUCGGCGGCCCGGCGGAUCUUUCCCGCCCCCCGUUCCUCCCGACCCCUCCACCCGCCCUCCCUUCCCCCGCCGCCCCUCCUCCUCCUCCCCGGAGGGGGCGGGCUCCGGCGGGUGCGGGGGUGGGCGGGCGGGGCCGGGGGUGGGGUCGGCGGGGGACCGUCCCCCGACCGGCGACCGGCCGCCGCCGGGCGCAUUUCCACCGCGGCGGUGCGCCGCGACCGGCUCCGGGACGGCUGGGAAGGCCCGGCGGGGAAGGUGGCUCGGGGGGCCCCGUCCGUCCGUCCGUCCGUCCUCCUCCUCCCCCGUCUCCGCCCCCCGGCCCCGCGUCCUCCCUCGGGAGGGCGCGCGGGUCGGGGCGGCGGCGGCGGCGGCGGUGGCGGCGGCGGCGGCGGCGGCGGGACCGAAACCCCCCCCGAGUGUUACAGCCCCCCCGGCAGCAGCACUCGCCGAAUCCCGGGGCCGAGGGAGCGAGACCCGUCGCCGCGCUCUCCCCCCUCCCGGCGCCCACCCCCGCGGGGAAUCCCCCGCGAGGGGGGUCUCCCCCGCGGGGGCGCGCCGGCGUCUCCUCGUGGGGGGGCCGGGCCACCCCUCCCACGGCGCGACCGCUCUCCCACCCCUCCUCCCCGCGCCCCCGCCCCGGCGACGGGGGGGGUGCCGCGCGCGGGUCGGGGGGCGGGGCGGACUGUCCCCAGUGCGCCCCGGGCGGGUCGCGCCGUCGGGCCCGGGGGAGGUUCUCUCGGGGCCACGCGCGCGUCCCCCGAAGAGGGGGACGGCGGAGCGAGCGCACGGGGUCGGCGGCGACGUCGGCUACCCACCCGACCCGUCUUGAAACACGGACCAAGGAGUCUAACACGUGCGCGAGUCGGGGGCUCGCACGAAAGCCGCCGUGGCGCAAUGAAGGUGAAGGCCGGCGCGCUCGCCGGCCGAGGUGGGAUCCCGAGGCCUCUCCAGUCCGCCGAGGGCGCACCACCGGCCCGUCUCGCCCGCCGCGCCGGGGAGGUGGAGCACGAGCGCACGUGUUAGGACCCGAAAGAUGGUGAACUAUGCCUGGGCAGGGCGAAGCCAGAGGAAACUCUGGUGGAGGUCCGUAGCGGUCCUGACGUGCAAAUCGGUCGUCCGACCUGGGUAUAGGGGCGAAAGACUAAUCGAACCAUCUAGUAGCUGGUUCCCUCCGAAGUUUCCCUCAGGAUAGCUGGCGCUCUCGCAGACCCGACGCACCCCCGCCACGCAGUUUUAUCCGGUAAAGCGAAUGAUUAGAGGUCUUGGGGCCGAAACGAUCUCAACCUAUUCUCAAACUUUAAAUGGGUAAGAAGCCCGGCUCGCUGGCGUGGAGCCGGGCGUGGAAUGCGAGUGCCUAGUGGGCCACUUUUGGUAAGCAGAACUGGCGCUGCGGGAUGAACCGAACGCCGGGUUAAGGCGCCCGAUGCCGACGCUCAUCAGACCCCAGAAAAGGUGUUGGUUGAUAUAGACAGCAGGACGGUGGCCAUGGAAGUCGGAAUCCGCUAAGGAGUGUGUAACAACUCACCUGCCGAAUCAACUAGCCCUGAAAAUGGAUGGCGCUGGAGCGUCGGGCCCAUACCCGGCCGUCGCCGGCAGUCGAGAGUGGACGGGAGCGGCGGGGGCGGCGCGCGCGCGCGCGCGUGUGGUGUGCGUCGGAGGGCGGCGGCGGCGGCGGCGGCGGGGGUGUGGGGUCCUUCCCCCGCCCCCCCCCCCACGCCUCCUCCCCUCCUCCCGCCCACGCCCCGCUCCCCGCCCCCGGAGCCCCGCGGACGCUACGCCGCGACGAGUAGGAGGGCCGCUGCGGUGAGCCUUGAAGCCUAGGGCGCGGGCCCGGGUGGAGCCGCCGCAGGUGCAGAUCUUGGUGGUAGUAGCAAAUAUUCAAACGAGAACUUUGAAGGCCGAAGUGGAGAAGGGUUCCAUGUGAACAGCAGUUGAACAUGGGUCAGUCGGUCCUGAGAGAUGGGCGAGCGCCGUUCCGAAGGGACGGGCGAUGGCCUCCGUUGCCCUCGGCCGAUCGAAAGGGAGUCGGGUUCAGAUCCCCGAAUCCGGAGUGGCGGAGAUGGGCGCCGCGAGGCGUCCAGUGCGGUAACGCGACCGAUCCCGGAGAAGCCGGCGGGAGCCCCGGGGAGAGUUCUCUUUUCUUUGUGAAGGGCAGGGCGCCCUGGAAUGGGUUCGCCCCGAGAGAGGGGCCCGUGCCUUGGAAAGCGUCGCGGUUCCGGCGGCGUCCGGUGAGCUCUCGCUGGCCCUUGAAAAUCCGGGGGAGAGGGUGUAAAUCUCGCGCCGGGCCGUACCCAUAUCCGCAGCAGGUCUCCAAGGUGAACAGCCUCUGGCAUGUUGGAACAAUGUAGGUAAGGGAAGUCGGCAAGCCGGAUCCGUAACUUCGGGAUAAGGAUUGGCUCUAAGGGCUGGGUCGGUCGGGCUGGGGCGCGAAGCGGGGCUGGGCGCGCGCCGCGGCUGGACGAGGCGCCGCCGCCCCCCCCACGCCCGGGGCACCCCCCUCGCGGCCCUCCCCCGCCCCACCCCGCGCGCGCCGCUCGCUCCCUCCCCGCCCCGCGCCCUCUCUCUCUCUCUCUCCCCCGCUCCCCGUCCUCCCCCCUCCCCGGGGGAGCGCCGCGUGGGGGCGGCGGCGGGGGGAGAAGGGUCGGGGCGGCAGGGGCCGGCGGCGGCCCGCCGCGGGGCCCCGGCGGCGGGGGCACGGUCCCCCGCGAGGGGGGCCCGGGCACCCGGGGGGCCGGCGGCGGCGGCGACUCUGGACGCGAGCCGGGCCCUUCCCGUGGAUCGCCCCAGCUGCGGCGGGCGUCGCGGCCGCCCCCGGGGAGCCCGGCGGGCGCCGGCGCGCCCCCCCCCCCACCCCACGUCUCGUCGCGCGCGCGUCCGCUGGGGGCGGGGAGCGGUCGGGCGGCGGCGGUCGGCGGGCGGCGGGGCGGGGCGGUUCGUCCCCCCGCCCUACCCCCCCGGCCCCGUCCGCCCCCCGUUCCCCCCUCCUCCUCGGCGCGCGGCGGCGGCGGCGGCAGGCGGCGGAGGGGCCGCGGGCCGGUCCCCCCCGCCGGGUCCGCCCCCGGGGCCGCGGUUCCGCGCGGCGCCUCGCCUCGGCCGGCGCCUAGCAGCCGACUUAGAACUGGUGCGGACCAGGGGAAUCCGACUGUUUAAUUAAAACAAAGCAUCGCGAAGGCCCGCGGCGGGUGUUGACGCGAUGUGAUUUCUGCCCAGUGCUCUGAAUGUCAAAGUGAAGAAAUUCAAUGAAGCGCGGGUAAACGGCGGGAGUAACUAUGACUCUCUUAAGGUAGCCAAAUGCCUCGUCAUCUAAUUAGUGACGCGCAUGAAUGGAUGAACGAGAUUCCCACUGUCCCUACCUACUAUCCAGCGAAACCACAGCCAAGGGAACGGGCUUGGCGGAAUCAGCGGGGAAAGAAGACCCUGUUGAGCUUGACUCUAGUCUGGCACGGUGAAGAGACAUGAGAGGUGUAGAAUAAGUGGGAGGCCCCCGGCGCCCCCCCGGUGUCCCCGCGAGGGGCCCGGGGCGGGGUCCGCCGGCCCUGCGGGCCGCCGGUGAAAUACCACUACUCUGAUCGUUUUUUCACUGACCCGGUGAGGCGGGGGGGCGAGCCCCGAGGGGCUCUCGCUUCUGGCGCCAAGCGCCCGGCCGCGCGCCGGCCGGGCGCGACCCGCUCCGGGGACAGUGCCAGGUGGGGAGUUUGACUGGGGCGGUACACCUGUCAAACGGUAACGCAGGUGUCCUAAGGCGAGCUCAGGGAGGACAGAAACCUCCCGUGGAGCAGAAGGGCAAAAGCUCGCUUGAUCUUGAUUUUCAGUACGAAUACAGACCGUGAAAGCGGGGCCUCACGAUCCUUCUGACCUUUUGGGUUUUAAGCAGGAGGUGUCAGAAAAGUUACCACAGGGAUAACUGGCUUGUGGCGGCCAAGCGUUCAUAGCGACGUCGCUUUUUGAUCCUUCGAUGUCGGCUCUUCCUAUCAUUGUGAAGCAGAAUUCACCAAGCGUUGGAUUGUUCACCCACUAAUAGGGAACGUGAGCUGGGUUUAGACCGUCGUGAGACAGGUUAGUUUUACCCUACUGAUGAUGUGUUGUUGCCAUGGUAAUCCUGCUCAGUACGAGAGGAACCGCAGGUUCAGACAUUUGGUGUAUGUGCUUGGCUGAGGAGCCAAUGGGGCGAAGCUACCAUCUGUGGGAUUAUGACUGAACGCCUCUAAGUCAGAAUCCCGCCCAGGCGGAACGAUACGGCAGCGCCGCGGAGCCUCGGUUGGCCUCGGAUAGCCGGUCCCCCGCCUGUCCCCGCCGGCGGGCCGCCCCCCCCCUCCACGCGCCCCGCGCGCGCGGGAGGGCGCGUGCCCCGCCGCGCGCCGGGACCGGGGUCCGGUGCGGAGUGCCCUUCGUCCUGGGAAACGGGGCGCGGCCGGAGAGGCGGCCGCCCCCUCGCCCGUCACGCACCGCACGUUCGUGGGGAACCUGGCGCUAAACCAUUCGUAGACGACCUGCUUCUGGGUCGGGGUUUCGUACGUAGCAGAGCAGCUCCCUCGCUGCGAUCUAUUGAAAGUCAGCCCUCGACACAAGGGUUUGUC';

rna();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count120 = 0; count120 < 4500; count120++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'UACCUGGUUGAUCCUGCCAGUAGCAUAUGCUUGUCUCAAAGAUUAAGCCAUGCAUGUCUGAGUACGCACGGCCGGUACAGUGAAACUGCGAAUGGCUCAUUAAAUCAGUUAUGGUUCCUUUGGUCGCUCGCUCCUCUCCUACUUGGAUAACUGUGGUAAUUCUAGAGCUAAUACAUGCCGACGGGCGCUGACCCCCUUCGCGGGGGGGAUGCGUGCAUUUAUCAGAUCAAAACCAACCCGGUCAGCCCCUCUCCGGCCCCGGCCGGGGGGCGGGCGCCGGCGGCUUUGGUGACUCUAGAUAACCUCGGGCCGAUCGCACGCCCCCCGUGGCGGCGACGACCCAUUCGAACGUCUGCCCUAUCAACUUUCGAUGGUAGUCGCCGUGCCUACCAUGGUGACCACGGGUGACGGGGAAUCAGGGUUCGAUUCCGGAGAGGGAGCCUGAGAAACGGCUACCACAUCCAAGGAAGGCAGCAGGCGCGCAAAUUACCCACUCCCGACCCGGGGAGGUAGUGACGAAAAAUAACAAUACAGGACUCUUUCGAGGCCCUGUAAUUGGAAUGAGUCCACUUUAAAUCCUUUAACGAGGAUCCAUUGGAGGGCAAGUCUGGUGCCAGCAGCCGCGGUAAUUCCAGCUCCAAUAGCGUAUAUUAAAGUUGCUGCAGUUAAAAAGCUCGUAGUUGGAUCUUGGGAGCGGGCGGGCGGUCCGCCGCGAGGCGAGCCACCGCCCGUCCCCGCCCCUUGCCUCUCGGCGCCCCCUCGAUGCUCUUAGCUGAGUGUCCCGCGGGGCCCGAAGCGUUUACUUUGAAAAAAUUAGAGUGUUCAAAGCAGGCCCGAGCCGCCUGGAUACCGCAGCUAGGAAUAAUGGAAUAGGACCGCGGUUCUAUUUUGUUGGUUUUCGGAACUGAGGCCAUGAUUAAGAGGGACGGCCGGGGGCAUUCGUAUUGCGCCGCUAGAGGUGAAAUUCUUGGACCGGCGCAAGACGGACCAGAGCGAAAGCAUUUGCCAAGAAUGUUUUCAUUAAUCAAGAACGAAAGUCGGAGGUUCGAAGACGAUCAGAUACCGUCGUAGUUCCGACCAUAAACGAUGCCGACCGGCGAUGCGGCGGCGUUAUUCCCAUGACCCGCCGGGCAGCUUCCGGGAAACCAAAGUCUUUGGGUUCCGGGGGGAGUAUGGUUGCAAAGCUGAAACUUAAAGGAAUUGACGGAAGGGCACCACCAGGAGUGGAGCCUGCGGCUUAAUUUGACUCAACACGGGAAACCUCACCCGGCCCGGACACGGACAGGAUUGACAGAUUGAUAGCUCUUUCUCGAUUCCGUGGGUGGUGGUGCAUGGCCGUUCUUAGUUGGUGGAGCGAUUUGUCUGGUUAAUUCCGAUAACGAACGAGACUCUGGCAUGCUAACUAGUUACGCGACCCCCGAGCGGUCGGCGUCCCCCAACUUCUUAGAGGGACAAGUGGCGUUCAGCCACCCGAGAUUGAGCAAUAACAGGUCUGUGAUGCCCUUAGAUGUCCGGGGCUGCACGCGCGCUACACUGACUGGCUCAGCGUGUGCCUACCCUACGCCGGCAGGCGCGGGUAACCCGUUGAACCCCAUUCGUGAUGGGGAUCGGGGAUUGCAAUUAUUCCCCAUGAACGAGGAAUUCCCAGUAAGUGCGGGUCAUAAGCUUGCGUUGAUUAAGUCCCUGCCCUUUGUACACACCGCCCGUCGCUACUACCGAUUGGAUGGUUUAGUGAGGCCCUCGGAUCGGCCCCGCCGGGGUCGGCCCACGGCCCUGGCGGAGCGCUGAGAAGACGGUCGAACUUGACUAUCUAGAGGAAGUAAAAGUCGUAACAAGGUUUCCGUAGGUGAACCUGCGGAAGGAUCAUUA';

rna();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count121 = 0; count121 < 4500; count121++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count122 = 0; count122 < 4500; count122++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count123 = 0; count123 < 4500; count123++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count124 = 0; count124 < 4500; count124++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count125 = 0; count125 < 4500; count125++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count126 = 0; count126 < 4500; count126++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count127 = 0; count127 < 4500; count127++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count128 = 0; count128 < 4500; count128++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count129 = 0; count129 < 4500; count129++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count130 = 0; count130 < 4500; count130++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

cobalamin();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count131 = 0; count131 < 4500; count131++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count132 = 0; count132 < 4500; count132++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count133 = 0; count133 < 4500; count133++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count134 = 0; count134 < 4500; count134++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count135 = 0; count135 < 4500; count135++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count136 = 0; count136 < 4500; count136++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count137 = 0; count137 < 4500; count137++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count138 = 0; count138 < 4500; count138++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count139 = 0; count139 < 4500; count139++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count140 = 0; count140 < 4500; count140++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

cobalamin();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count141 = 0; count141 < 4500; count141++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count142 = 0; count142 < 4500; count142++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count143 = 0; count143 < 4500; count143++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count144 = 0; count144 < 4500; count144++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count145 = 0; count145 < 4500; count145++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madsrdpasd qmqhwkeqra aqkadvlttg agnpvgdkln vitvgprgpl lvqdvvftde 61 mahfdrerip ervvhakgag afgyfevthd itkyskakvf ehigkktpia vrfstvages 121 gsadtvrdpr gfavkfyted gnwdlvgnnt piffirdpil fpsfihsqkr npqthlkdpd 181 mvwdfwslrp eslhqvsflf sdrgipdghr hmngygshtf klvnangeav yckfhyktdq 241 giknlsveda arlsqedpdy girdlfnaia tgkypswtfy iqvmtfnqae tfpfnpfdlt 301 kvwphkdypl ipvgklvlnr npvnyfaeve qiafdpsnmp pgieaspdkm lqgrlfaypd 361 thrhrlgpny lhipvncpyr arvanyqrdg pmcmqdnqgg apnyypnsfg apeqqpsale 421 hsiqysgevr rfntanddnv tqvrafyvnv lneeqrkrlc eniaghlkda qifiqkkavk 481 nftevhpdyg shiqalldky naekpknaih tfvqsgshla arekanl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count146 = 0; count146 < 4500; count146++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count147 = 0; count147 < 4500; count147++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count148 = 0; count148 < 4500; count148++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count149 = 0; count149 < 4500; count149++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count150 = 0; count150 < 4500; count150++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

cobalamin();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count151 = 0; count151 < 4500; count151++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mperdsepfs nplapdghdv ddphsfhqsk ltnedfrkll mtpraaptsa ppsksrhhem 61 preynededp aarrrkkksy yaklrqqeie rerelaekyr drakerrdgv nkdyeeteli 121 sttanyravg ptaeadksaa ekrrqliqes kflggdmeht hlvkgldfal lqkvraeias 181 kekeeeelme kpqketkkde dpenkiefkt rlgrnvyrml fkskayerne lflpgrmayv 241 vdlddeyadt dipttlirsk adcptmeaqt tlttndivis kltqilsylr qgtrnkklkk 301 kdkgkleekk ppeadmnife digdyvpstt ktprdkerer yrererdrer drdrdrerer 361 erdrererer drereeekkr hsyfekpkvd depmdvdkgp gstkeliksi nekfagsagw 421 egteslkkpe dkkqlgdffg msnsyaecyp atmddmavds deevdyskmd qgnkkgplgr 481 wdfdtqeeys eymnnkealp kaafqygikm segrktrrfk etndkaeldr qwkkisaiie 541 krkkmeadgv evkrpky';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count152 = 0; count152 < 4500; count152++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 marnaekamt alarfrqaql eegkvkerrp flasectelp kaekwrrqii geiskkvaqi 61 qnaglgefri rdlndeinkl lrekghwevr ikelggpdyg kvgpkmldhe gkevpgnrgy 121 kyfgaakdlp gvrelfeker qvrwlmpvip alweaeaggs qalppprktr aelmkaidfe 181 yygyldeddg vivpleqeye kklraelvek wkaerearla rgekeeeeee eeeiniyavt 241 eeesdeegsq ekggddsqqk fiahvpvpsq qeieealvrr kkmellqkya setlqaqsee 301 arrllgy';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count153 = 0; count153 < 4500; count153++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msvpsalmkq ppiqstagav pvrnekgeis mekvkvkryv sgkrpdyapm essdeedeef 61 qfikkakeqe aepeeqeeds ssdprlrrlq nrisedveer larhrkivep evvgesdsev 121 egdawrmere dsseeeeeei ddeeierrrg mmrqraqerk neemevmeve degrsgeese 181 seseyeeytd sedemeprlk pvfirkkdrv tvqereaeal kqkeleqeak rmaeerrkyt 241 lkiveeetkk eleenkrsla aldalntdde ndeeeyeawk vrelkrikrd redrealeke 301 kaeiermrnl teeerraelr angkvitnka vkgkykflqk yyhrgaffmd edeevykrdf 361 saptledhfn ktilpkvmqv knfgrsgrtk ythlvdqdtt sfdsawgqes aqntkffkqk 421 aagvrdvfer psakkrktt';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count154 = 0; count154 < 4500; count154++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mattkrvlyv gglaeevddk vlhaafipfg ditdiqipld yetekhrgfa fvefelaeda 61 aaaidnmnes elfgrtirvn lakpmrikeg ssrpvwsddd wlkkfsgktl eenkeeegse 121 ppkaetqege piakkarsnp qvymdikign kpagriqmll rsdvvpmtae nfrclcthek 181 gfgfkgssfh riipqfmcqg gdftnhngtg gksiygkkfd denfilkhtg pgllsmansg 241 pntngsqffl tcdktdwldg khvvfgevte gldvlrqiea qgskdgkpkq kviiadcgey 301 v';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count155 = 0; count155 < 4500; count155++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaippdswq ppnvyletsm giivlelywk hapktcknfa elarrgyyng tkfhriikdf 61 miqggdptgt grggasiygk qfedelhpdl kftgagilam anagpdtngs qffvtlaptq 121 wldgkhtifg rvcqgigmvn rvgmvetnsq drpvddvkii kaypsg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count156 = 0; count156 < 4500; count156++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mplpvalqtr lakrgilkhl epepeeeiia edydddpvdy eatrleglpp swykvfdpsc 61 glpyywnadt dlvswlsphd pnsvvtksak klrssnadae ekldrshdks drghdksdrs 121 hekldrghdk sdrghdksdr drergydkvd rererdrerd rdrgydkadr eegkerrhhr 181 reelapypks kkavsrkdee ldpmdpssys daprgtwstg lpkrneaktg adttaagplf 241 qqrpypspga vlranaeasr tkqqd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count157 = 0; count157 < 4500; count157++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 manrtvkdah sihgtnpqyl vekiirtriy eskywkeecf gltaelvvdk amelrfvggv 61 yggnikptpf lcltlkmlqi qpekdiivef iknedfkyvr mlgalymrlt gtaidcykyl 121 eplyndyrki ksqnrngefe lmhvdefide llhservcdi ilprlqkryv leeaeqlepr 181 vsaleedmdd vesseeeeee deklervpsp dhrrrsyrdl dkprrsptlr yrrsrsrspr 241 rrsrspkrrs psprrerhrs ksprrhrsrs rdrrhrsrsk spghhrshrh rshskspers 301 kkshkksrrg ne';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count158 = 0; count158 < 4500; count158++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matslgsnty nrqnwedadf pilcqtclge npyirmtkek ygkeckicar pftvfrwcpg 61 vrmrfkktev cqtcsklknv cqtclldley glpiqvrdag lsfkddmpks dvnkeyytqn 121 mereisnsdg trpvgmlgka tstsdmllkl arttpyykrn rphicsfwvk geckrgeecp 181 yrhekptdpd dpladqnikd ryygindpva dkllkrastm prldppedkt ittlyvgglg 241 dtitetdlrn hfyqfgeirt itvvqrqqca fiqfatrqaa evaaeksfnk livngrrlnv 301 kwgrsqaarg kekekdgttd sgiklepvpg lpgalppppa aeeeasanyf nlppsgppav 361 vnialppppg iapppppgfg phmfhpmgpp ppfmrapgpi hypsqdpqrm gahagkhssp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count159 = 0; count159 < 4500; count159++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msieiessdv irlimqylke nslhralatl qeettvslnt vdsiesfvad insghwdtvl 61 qaiqslklpd ktlidlyeqv vlelielrel gaarsllrqt dpmimlkqtq peryihlenl 121 larsyfdpre aypdgsskek rraaiaqala gevsvvppsr lmallgqalk wqqhqgllpp 181 gmtidlfrgk aavkdveeek fptqlsrhik fgqkshveca rfspdgqylv tgsvdgfiev 241 wnfttgkirk dlkyqaqdnf mmmddavlcm cfsrdtemla tgaqdgkikv wkiqsgqclr 301 rferahskgv tclsfskdss qilsasfdqt irihglksgk tlkefrghss fvneatftqd 361 ghyiisassd gtvkiwnmkt tecsntfksl gstagtditv nsvillpknp ehfvvcnrsn 421 tvvimnmqgq ivrsfssgkr eggdfvccal sprgewiycv gedfvlycfs tvtgklertl 481 tvhekdvigi ahhphqnlia tysedgllkl wkp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count160 = 0; count160 < 4500; count160++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maltsflpap tqlsqdqlea eekarsqrsr qtslvssrre pppygyrkgw iprlledfgd 61 ggafpeihva qypldmgrkk kmsnalaiqv dsegkikyda iarqgqskdk viyskytdlv 121 pkevmnaddp dlqrpdeeai keitektrva leksvsqkva aampvraadk lapaqyiryt 181 psqqgvafns gakqrvirmv emqkdpmepp rfkinkkipr gppsppapvm hspsrkmtvk 241 eqqewkippc isnwknakgy tipldkrlaa dgrglqtvhi nenfaklaea lyiadrkare 301 avemraqver kmaqkekekh eeklremaqk arerragikt hvekedgear erdeirhdrr 361 kerqhdrnls raapdkrskl qrnenrdise vialgvpnpr tsnevqydqr lfnqskgmds 421 gfaggedeiy nvydqawrgg kdmaqsiyrp sknldkdmyg ddleariktn rcqaiqlnfs 481 vytgeskvvh sffsfsldlf ptrsflvqtv dreaekdqcs lrkillvwts fwkkpnsmva 541 lkdpqiaaap rntsmkarrg grnrhrslqs e';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count161 = 0; count161 < 4500; count161++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maltsflpap tqlsqdqlea eekarsqrsr qtslvssrre pppygyrkgw iprlledfgd 61 ggafpeihva qypldmgrkk kmsnalaiqv dsegkikyda iarqgqskdk viyskytdlv 121 pkevmnaddp dlqrpdeeai keitektrva leksvsqkva aampvraadk lapaqyiryt 181 psqqgvafns gakqrvirmv emqkdpmepp rfkinkkipr gppsppapvm hspsrkmtvk 241 eqqewkippc isnwknakgy tipldkrlaa dgrglqtvhi nenfaklaea lyiadrkare 301 avemraqver kmaqkekekh eeklremaqk arerragikt hvekedgear erdeirhdrr 361 kerqhdrnls raapdkrskl qrnenrdise vialgvpnpr tsnevqydqr lfnqskgmds 421 gfaggedeiy nvydqawrgg kdmaqsiyrp sknldkdmyg ddleariktn rcqaiqlnfs 481 vytgeskvvh sffsfsldlf ptrsflvqtv dreaekdqcs lrkillvwts fwkkpnsmva 541 lkdpqiaaap rntsmkarrg grnrhrslqs e';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count162 = 0; count162 < 4500; count162++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mslshlyrdg egriddddde renfeitdwd lqnefnpnrq rhwqtkeeat ygvwaerdsd 61 derpsfggkr ardysapvnf isaglkkgaa eeaeledsdd eekpvkqddf pkdfgprklk 121 tggnfkpsqk gfaggtksfm dfgswerhtk gigqkllqkm gyvpgrglgk naqgiinpie 181 akqrkgkgav gaygserttq smqdfpvvds eeeaeeefqk elsqwrkdps gskkkpkysy 241 ktveelkakg riskkltapq kelsqvkvid mtgreqkvyy sysqishkhn vpddglplqs 301 qqlpqsgkea kapgfalpel ehnlqllidl teqeiiqndr qlqyerdmvv nlfhelekmt 361 evldheervi snlskvlemv eecerrmqpd csnpltldec arifetlqdk yyeeyrmsdr 421 vdlavaivyp lmkeyfkewd plkdctygte iiskwkslle ndqllshggq dlsadafhrl 481 iwevwmpfvr nivtqwqprn cdpmvdflds wvhiipvwil dnildqlifp klqkevenwn 541 pltdtvpihs wihpwlplmq arleplyspi rsklssalqk whpsdssakl ilqpwkdvft 601 pgsweafmvk nivpklgmcl gelvinphqq hmdafywvid wegmisvssl vgllekhffp 661 kwlqvlcswl snspnyeeit kwylgwksmf sdqvlahpsv kdkfnealdi mnravssnvg 721 aymqpgaren iaylthterr kdfqyeamqe rreaenmaqr gigvaassvp mnfkdlietk 781 aeehnivfmp vigkrhegkq lytfgriviy idrgvvfvqg ektwvptslq slidmak';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count163 = 0; count163 < 4500; count163++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madywksqpk kfcdyckcwi adnrpsvefh ergknhkenv akriseikqk sldkakeeek 61 askefaamea aalkayqedl krlgleseil epsitpvtst ipptstsnqq kekkekkkrk 121 kdpskgrwve gitsegyhyy ydlisgasqw ekpegfqgdl kktavktvwv eglsedgfty 181 yyntetgesr wekpddfiph tsdlpsskvn enslgtldes kssdshsdsd geqeaeeggv 241 stetekpkik fkeknknsdg gsdpetqkek siqkqnslgs neeksktlkk snpygewqei 301 kqevesheev dlelpstene yvstseadgg gepkvvfkek tvtslgvmad gvapvfkkrr 361 tengksrnlr qrgddq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count164 = 0; count164 < 4500; count164++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrrstsstk sgkfmnptdq arkearkrel kknkkqrmmv raavlkmkdp kqiirdmekl 61 demefnpvqq pqlnekvlkd krkklretfe rilrlyeken pdiykelrkl eveyeqkraq 121 lsqyfdavkn aqhvevesip lpdmphapsn iliqdiplpg aqppsilkkt saygpptrav 181 silpllghgv prlppgrkpp gpppgppppq vvqmygrkvg faldlpprrr dedmlyspel 241 aqrghdddvs stseddgype dmdqdkhdds tddsdtdksd gesdgdefvh rdngerdnne 301 ekksglsvrf admpgksrkk kknmkeltpl qammlrmagq eipeegreve efsedddedd 361 sddseaekqs qkqhkeeshs dgtstassqq qappqsvpps qiqappmpgp pplgpppapp 421 lrppgpptgl ppgpppgapp flrppgmpgl rgplprllpp gpppgrppgp ppgpppglpp 481 gppprgpppr lpppappgip pprpgmmrpp lvpplgpapp glfppaplpn pgvlsappnl 541 iqrpkaddts aatiekkata tisakpqitn pkaeitrfvp talrvrrenk gataapqrks 601 eddsavplak aapksgpsvp vsvqtkddvy eafmkemegl l';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count165 = 0; count165 < 4500; count165++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvvmarlsrp erpdlvfeee dlpyeeeimr nqfsvkcwlr yiefkqgapk prlnqlyera 61 lkllpcsykl wyrylkarra qvkhrcvtdp ayedvnnche rafvfmhkmp rlwldycqfl 121 mdqgrvthtr rtfdralral pitqhsriwp lylrflrshp lpetavrgyr rflklspesa 181 eeyieylkss drldeaaqrl atvvnderfv skagksnyql whelcdlisq npdkvqslnv 241 daiirggltr ftdqlgklwc sladyyirsg hfekardvye eairtvmtvr dftqvfdsya 301 qfeesmiaak metaselgre eeddvdlelr larfeqlisr rplllnsvll rqnphhvhew 361 hkrvalhqgr preiintyte avqtvdpfka tgkphtlwva fakfyedngq lddarvilek 421 atkvnfkqvd dlasvwcqcg elelrhenyd ealrllrkat alparraeyf dgsepvqnrv 481 ykslkvwsml adleeslgtf qstkavydri ldlriatpqi vinyamflee hkyfeesfka 541 yergislfkw pnvsdiwsty ltkfiarygg rklerardlf eqaldgcppk yaktlyllya 601 qleeewglar hamavyerat ravepaqqyd mfniyikraa eiygvthtrg iyqkaievls 661 deharemclr fadmecklge idraraiysf csqicdprtt gafwqtwkdf evrhgnedti 721 kemlrirrsv qatyntqvnf masqmlkvsg satgtvsdla pgqsgmddmk lleqraeqla 781 aeaerdqplr aqskilfvrs dasreelael aqqvnpeeiq lgedededem dlepnevrle 841 qqsvpaavfg slked';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count166 = 0; count166 < 4500; count166++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 masgsgtknl dfrrkwdkde yeklaekrlt eerekkdgkp vqpvkrellr hrdykvdles 61 klgktivitk ttpqsemggy ycnvcdcvvk dsinfldhin gkkhqrnlgm smrverstld 121 qvkkrfevnk kkmeekqkdy dfeermkelr eeeekakayk kekqkekkrr aeedltfeed 181 demaavmgfs gfgstkksy';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count167 = 0; count167 < 4500; count167++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqdgrkggay agkmeattag vgrleeealr rkerlkalre ktgrkdkedg epktkhlree 61 eeegekhrel rlrnyvpede dlkkrrvpqa kpvaveekvk eqleaakpep vieevdlanl 121 aprkpdwdlk rdvakklekl kkrtqraiae lirerlkgqe dslasavdaa teqktcdsd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count168 = 0; count168 < 4500; count168++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkssvaqikp ssghdrrenl nsyqrnsspe dryeeqersp rdrdyfdysr sdyehsrrgr 61 sydssmesrn rdrekrrere rdtdrkrsrk spspgrrnpe tsvtqsssaq depatkkkkd 121 eldplltrtg gayippaklr mmqeqitdkn slayqrmswe alkksingli nkvnisnisi 181 iiqellqeni vrgrgllsrs vlqaqsaspi fthvyaalva iinskfpqig elilkrliln 241 frkgyrrndk qlcltaskfv ahlinqnvah evlclemltl llerptddsv evaigflkec 301 glkltqvspr ginaiferlr nilheseidk rvqymievmf avrkdgfkdh piilegldlv 361 eeddqfthml pleddynped vlnvfkmdpn fmeneekyka ikkeildegd tdsntdqdag 421 sseedeeeee eegeedeegq kvtihdktei nlvsfrrtiy laiqssldfe ecahkllkme 481 fpesqtkelc nmildccaqq rtyekffgll agrfcmlkke ymesfegifk eqydtihrle 541 tnklrnvakm fahllytdsl pwsvlecikl seetttsssr ifvkiffqel ceymglpkln 601 arlkdetlqp ffegllprdn prntrfainf ftsiglgglt delrehlknt pkvivaqkpd 661 veqnksspss sssasssses dssdsdsdss dsssesssee sdsssisshs sasandvrkk 721 ghgktrskev dklirnqqtn drkqkerrqe hghqetrter errsekhrdq nssgsnwrdp 781 itkytsdkdv psernnysrv andrdqemhi dlenkhgdpk kkrgerrnsf senekhthri 841 kdsenfrrkd rskskemnrk hsgsrsdedr yqngaerrwe kssryseqsr eskknqdrrr 901 ekspakqk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count169 = 0; count169 < 4500; count169++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mgggdlnlkk swhpqtlrnv ekvwkaeqkh eaerkkieel qrelreerar eemqryaedv 61 gavkkkeekl dwmyqgpggm vnrdeyllgr pidkyvfekm eekeagcsse tgllpgsifa 121 psganslldm askiredplf iirkkeeekk revlnnpvkm kkikellqms lekkekkkkk 181 ekkkkhkkhk hrssssdrss sedehsagrs qkkmansspv lskvpgyglq vrnsdrnqgl 241 qgpltaeqkr ghgmknhsrs rssshspprh askkstreag srdrrsrslg rrsrsprpsk 301 lhnskvnrre tgqtrspspk kevyqrrhap gytrklsaee lerkrqemme nakwreeerl 361 nilkrhakde ereqrlekld srdgkfihrm klesastssl edrvkrniys lqrtsvalek 421 nfmkr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count170 = 0; count170 < 4500; count170++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msniyiqepp tngkvllktt agdidielws keapkacrnf iqlcleayyd ntifhrvvpg 61 fivqggdptg tgsggesiyg apfkdefhsr lrfnrrglva managshdng sqffftlgra 121 delnnkhtif gkvtgdtvyn mlrlsevdid dderphnphk ikscevlfnp fddiipreik 181 rlkkekpeee vkklkpkgtk nfsllsfgee aeeeeeevnr vsqsmkgksk sshdllkddp 241 hlssvpvves ekgdapdlvd dgedesaehd eyidgdeknl mreriakklk kdtsanvksa 301 gegevekksv srseelrkea rqlkrellaa kqkkvenaak qaekrseeee appdgavaey 361 rrekqkyeal rkqqskkgts redqtlalln qfkskltqai aetpendipe teveddegwm 421 shvlqfedks rkvkdasmqd sdtfeiydpr npvnkrrree skklmrekke rr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count171 = 0; count171 < 4500; count171++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdtdlydefg nyigpeldsd edddelgret kdldemdddd ddddvgdhdd dhpgmevvlh 61 edkkyyptae evygpeveti vqeedtqplt epiikpvktk kftlmeqtlp vtvyemdfla 121 dlmdnselir nvtlcghlhh gktcfvdcli eqthpeirkr ydqdlcytdi lfteqergvg 181 ikstpvtvvl pdtkgksylf nimdtpghvn fsdevtaglr isdgvvlfid aaegvmlnte 241 rlikhavqer lavtvcinki drlilelklp ptdayyklrh ivdevnglis mystdenlil 301 spllgnvcfs ssqysicftl gsfakiyadt fgdinyqefa krlwgdiyfn pktrkftkka 361 ptsssqrsfv efileplyki laqvvgdvdt slprtldelg ihltkeelkl nirpllrlvc 421 kkffgeftgf vdmcvqhips pkvgakpkie htytggvdsd lgeamsdcdp dgplmchttk 481 mystddgvqf hafgrvlsgt ihagqpvkvl genytledee dsqictvgrl wisvaryhie 541 vnrvpagnwv liegvdqpiv ktatiteprg neeaqifrpl kfnttsviki avepvnpsel 601 pkmldglrkv nksypslttk veesgehvil gtgelyldcv mhdlrkmyse idikvadpvv 661 tfcetvvets slkcfaetpn kknkitmiae plekglaedi enevvqitwn rkklgeffqt 721 kydwdllaar siwafgpdat gpnilvddtl psevdkallg svkdsivqgf qwgtregplc 781 delirnvkfk ildavvaqep lhrgggqiip tarrvvysaf lmatprlmep yyfvevqapa 841 dcvsavytvl arrrghvtqd apipgsplyt ikafipaids fgfetdlrth tqgqafslsv 901 fhhwqivpgd pldksivirp lepqpaphla refmiktrrr kglsedvsis kffddpmlle 961 lakqdvvlny pm';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count172 = 0; count172 < 4500; count172++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mattatmats gsarkrllke edmtkvefet seevdvtptf dtmglredll rgiyaygfek 61 psaiqqraik qiikgrdvia qsqsgtgkta tfsisvlqcl diqvretqal ilaptrelav 121 qiqkgllalg dymnvqchac iggtnvgedi rkldygqhvv agtpgrvfdm irrrslrtra 181 ikmlvldead emlnkgfkeq iydvyrylpp atqvvlisat lpheilemtn kfmtdpiril 241 vkrdeltleg ikqffvaver eewkfdtlcd lydtltitqa vifcntkrkv dwltekmrea 301 nftvssmhgd mpqkeresim kefrsgasrv listdvwarg ldvpqvslii nydlpnnrel 361 yihrigrsgr ygrkgvainf vknddirilr dieqyystqi dempmnvadl i';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count173 = 0; count173 < 4500; count173++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maardsdsee dlvsygtgle pleegerpkk piplqdqtvr dekgrykrfh gafsggfsag 61 yfntvgskeg wtpstfvssr qnradksvlg pedfmdeedl sefgiapkai vttddfaskt 121 kdrirekarq laaatapipg atllddlitp aklsvgfell rkmgwkegqg vgprvkrrpr 181 rqkpdpgvki ygcalppgss egsegedddy lpdnvtfapk dvtpvdftpk dnvhglaykg 241 ldphqalfgt sgehfnlfsg gseragdlge iglnkgrklg isgqafgvga leeedddiya 301 tetlskydtv lkdeepgdgl ygwtaprqyk nqkesekdlr yvgkildgfs laskplsskk 361 iypppelprd yrpvhyfrpm vaatsenshl lqvlsesagk atpdpgthsk hqlnaskrae 421 llgetpiqgs atsvleflsq kdkerikemk qatdlkaaql karslaqnaq ssraqlspaa 481 aaghcswnma lgggtatlka snfkpfakdp ekqkrydefl vhmkqgqkda lercldpsmt 541 ewergrerde faraallyas shstlssrft hakeeddsdq vevprdqend vgdkqsavkm 601 kmfgkltrdt fewhpdkllc krfnvpdpyp dstlvglprv krdkysvfnf ltlpetaslp 661 ttqassekvs qhrgpdksrk psrwdtskhe kkedsisefl slarskaepp kqqssplvnk 721 eeehapelsa nqtvnkdvda qaegegsrps mdlfraifas ssdeksssse deqgdseddq 781 agsgeanfqs sqdtdlgets svahalvpap qepppsfpiq kmqidereef gprlppvfcp 841 narqtlevpq kekhkknkdk hkakkehrrk kekkkkhrkh khkgkqknkk pekssssess 901 dssdsqsdee tadvspqell rrlkslplrr q';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count174 = 0; count174 < 4500; count174++) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = '1 mgkgvgrdky epaavseqgd kkgkkgkkdr dmdelkkevs mddhklslde lhrkygtdls 61 rgltsaraae ilardgpnal tpppttpewi kfcrqlfggf smllwigail cflaysiqaa 121 teeepqndnl ylgvvlsavv iitgcfsyyq eaksskimes fknmvpqqal virngekmsi 181 naeevvvgdl vevkggdrip adlriisang ckvdnssltg esepqtrspd ftnenpletr 241 niaffstncv egtargivvy tgdrtvmgri atlasglegg qtpiaaeieh fihiitgvav 301 flgvsffils lileytwlea vifligiiva nvpegllatv tvcltltakr marknclvkn 361 leavetlgst sticsdktgt ltqnrmtvah mwfdnqihea dttenqsgvs fdktsatwla 421 lsriaglcnr avfqanqenl pilkravagd asesallkci elccgsvkem reryakivei 481 pfnstnkyql sihknpntse pqhllvmkga perildrcss illhgkeqpl deelkdafqn 541 aylelgglge rvlgfchlfl pdeqfpegfq fdtddvnfpi dnlcfvglis midppraavp 601 davgkcrsag ikvimvtgdh pitakaiakg vgiisegnet vediaarlni pvsqvnprda 661 kacvvhgsdl kdmtseqldd ilkyhteivf artspqqkli ivegcqrqga ivavtgdgvn 721 dspalkkadi gvamgiagsd vskqaadmil lddnfasivt gveegrlifd nlkksiaytl 781 tsnipeitpf lifiianipl plgtvtilci dlgtdmvpai slayeqaesd imkrqprnpk 841 tdklvnerli smaygqigmi qalggfftyf vilaengflp ihllglrvdw ddrwindved 901 sygqqwtyeq rkiveftcht affvsivvvq wadlvicktr rnsvfqqgmk nkilifglfe 961 etalaaflsy cpgmgvalrm yplkptwwfc afpysllifv ydevrkliir rrpggwveke 1021 tyy';

} else {

peptidesequence = ' 1 margkakeeg swkkfiwnse kkeflgrtgg swfkillfyv ifygclagif igtiqvmllt 61 isefkptyqd rvappgltqi pqiqkteisf rpndpksyea yvlnivrfle kykdsaqrdd 121 mifedcgdvp sepkergdfn hergerkvcr fklewlgncs glndetygyk egkpciiikl 181 nrvlgfkpkp pknesletyp vmkynpnvlp vqctgkrded kdkvgnveyf glgnspgfpl 241 qyypyygkll qpkylqplla vqftnltmdt eirieckayg enigysekdr fqgrfdvkie 301 vks';

}

protein();

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

}

function vital\_components2() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count2 = 0; count2 < 4500; count2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count4 = 0; count4 < 4500; count4++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count6 = 0; count6 < 4500; count6++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count8 = 0; count8 < 4500; count8++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count10 = 0; count10 < 4500; count10++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count12 = 0; count12 < 4500; count12++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count14 = 0; count14 < 4500; count14++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count16 = 0; count16 < 4500; count16++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count18 = 0; count18 < 4500; count18++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count20 = 0; count20 < 4500; count20++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count22 = 0; count22 < 4500; count22++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count24 = 0; count24 < 4500; count24++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count26 = 0; count26 < 4500; count26++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count27 = 0; count27 < 4500; count27++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count28 = 0; count28 < 4500; count28++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count29 = 0; count29 < 4500; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count30 = 0; count30 < 4500; count30++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepsskkltg rlmlavggav lgslqfgynt gvinapqkvi eefynqtwvh rygesilptt 61 lttlwslsva ifsvggmigs fsvglfvnrf grrnsmlmmn llafvsavlm gfsklgksfe 121 mlilgrfiig vycglttgfv pmyvgevspt alrgalgtlh qlgivvgili aqvfgldsim 181 gnkdlwplll siifipallq civlpfcpes prfllinrne enraksvlkk lrgtadvthd 241 lqemkeesrq mmrekkvtil elfrspayrq piliavvlql sqqlsginav fyystsifek 301 agvqqpvyat igsgivntaf tvvslfvver agrrtlhlig lagmagcail mtialalleq 361 lpwmsylsiv aifgfvaffe vgpgpipwfi vaelfsqgpr paaiavagfs nwtsnfivgm 421 cfqyveqlcg pyvfiiftvl lvlffiftyf kvpetkgrtf deiasgfrqg gasqsdktpe 481 elfhplgads qv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count31 = 0; count31 < 4500; count31++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaskkavlg plvgavdqgt sstrflvfns ktaellshhq veikqefpre gwveqdpkei 61 lhsvyeciek tceklgqlni disnikaigv snqrettvvw dkitgeplyn avvwldlrtq 121 stveslskri pgnnnfvksk tglplstyfs avklrwlldn vrkvqkavee kralfgtids 181 wliwsltggv nggvhctdvt nasrtmlfni hslewdkqlc effgipmeil pnvrssseiy 241 glmkishsvk agalegvpis gclgdqsaal vgqmcfqigq akntygtgcf llcntghkcv 301 fsdhgllttv ayklgrdkpv yyalegsvai agavirwlrd nlgiiktsee ieklakevgt 361 sygcyfvpaf sglyapywep sargiicglt qftnkchiaf aaleavcfqt reildamnrd 421 cgiplshlqv dggmtsnkil mqlqadilyi pvvkpsmpet talgaamaag aaegvgvwsl 481 epedlsavtm erfepqinae eseirystwk kavmksmgwv ttqspesgip';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count32 = 0; count32 < 4500; count32++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeklriciv gsgnwgstia kiiginaanf snfedrvtmy vyeeiingkk lteiinethe 61 nvkylpghkl ppniiaipdv veaakdadil tfvvphqfik ricsalfgki kptaiglsli 121 kgfdkkqggg ielishiisk qlhipvsvlm ganlasevan emfcettigc kdknmapilk 181 dlmetsyfkv vvvedvdsve ccgalkniva cgagfidglg lgdntkaavm rlglmeiikf 241 vniffpggkk ttffescgva dliatcyggr nrkiceafvk tgkkiselek emlngqklqg 301 pftaeevnym lkaknmenrf plfttvhric igetmpmeli enlrnhpeyi detrnyqeck 361 csi';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count33 = 0; count33 < 4500; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfprektwni sfagcgflgv yyvgvasclr ehapflvana thiygasaga ltatalvtgv 61 clgeagakfi evskearkrf lgplhpsfnl vkiirsfllk vlpadsheha sgrlgisltr 121 vsdgenviis hfnskdeliq anvcsgfipv ycglippslq gvryvdggis dnlplyelkn 181 titvspfsge sdicpqdsst nihelrvtnt siqfnlrnly rlskalfppe plvlremckq 241 gyrdglrflq rngllnrpnp llalpparph gpedkdqave saqaedysql pgedhilehl 301 parlnealle acveptdllt tlsnmlpvrl atammvpytl plesalsfti rllewlpdvp 361 edirwmkeqt gsicqylvmr akrklgrhlp srlpeqvelr rvqslpsvpl scaayrealp 421 gwmrnnlslg dalakweecq rqlllglfct nvafppealr mrapadpapa padpaspqhq 481 lagpapllst papearpvig algl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count34 = 0; count34 < 4500; count34++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mepgsksvsr sdwqpephqr pitplepgpe ktpiaqpesk tlqgsntqqk pasnqrpltq 61 qetpaqhdae sqkepraqqk sasqeeflap qkpapqqspy iqrvlltqqe aasqqgpglg 121 kesitqqepa lrqrhvaqpg pgpgepppaq qeaestpaaq akpgakreps aptestsqet 181 peqsdkqttp vqgakskqgs ltelgfltkl qelsiqrsal ewkalsewvt dsesesdvgs 241 ssdtdspatm ggmvaqgvkl gfkgksgykv msgysgtsph ektsarnhrh yqdtasrlih 301 nmdlrtmtqs lvtlaednia ffssqgpget aqrlsgvfag vreqalglep algrllgvah 361 lfdldpetpa ngyrslvhta rcclahllhk sryvasnrrs iffrtshnla eleaylaalt 421 qlralvyyaq rllvtnrpgv lffegdeglt adflreyvtl hkgcfygrcl gfqftpairp 481 flqtisiglv sfgehykrne tglsvaassl ftsgrfaidp elrgaeferi tqnldvhfwk 541 afwnitemev lsslanmasa tvrvsrllsl ppeafemplt adptltvtis pplahtgpgp 601 vlvrlisydl regqdseels sliksngqrs lelwprpqqa prsrslivhf hgggfvaqts 661 rshepylksw aqelgapiis idyslapeap fpraleecff aycwaikhca llgstgeric 721 lagdsaggnl cftvalraaa ygvrvpdgim aaypatmlqp aaspsrllsl mdpllplsvl 781 skcvsayaga ktedhsnsdq kalgmmglvr rdtalllrdf rlgasswlns flelsgrksq 841 kmsepiaepm rrsvseaala qpqgplgtds lknltlrdls lrgnsetssd tpemslsaet 901 lspstpsdvn fllppedage eaeaknelsp mdrglgvraa fpegfhprrs sqgatqmply 961 sspivknpfm spllapdsml kslppvhiva caldpmldds vmlarrlrnl gqpvtlrvve 1021 dlphgfltla alcretrqaa elcverirlv ltppagagps getgaagvdg gcggrh';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count35 = 0; count35 < 4500; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfprektwni sfagcgflgv yyvgvasclr ehapflvana thiygasaga ltatalvtgv 61 clgeagakfi evskearkrf lgplhpsfnl vkiirsfllk vlpadsheha sgrlgisltr 121 vsdgenviis hfnskdeliq anvcsgfipv ycglippslq gvryvdggis dnlplyelkn 181 titvspfsge sdicpqdsst nihelrvtnt siqfnlrnly rlskalfppe plvlremckq 241 gyrdglrflq rngllnrpnp llalpparph gpedkdqave saqaedysql pgedhvlehl 301 parlnealle acveptdllt tlsnmlpvrl atammvpytl plesalsfti cllewlpdvp 361 edirwmkeqt gsicqylvmr akrklgrhlp srlpeqvelr rvqslpsvpl scaayreapp 421 gwmrnnlslg dalakweecq rqlllglfct nvafppealr mrapadpapa padpaspqhq 481 lagpapllst papearpvig algl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count36 = 0; count36 < 4500; count36++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 metgpedpss mpeessprrt pqsipyqdlp hlvnadgqyl fcrywkptgt pkalifvshg 61 agehsgryee larmlmgldl lvfahdhvgh gqsegermvv sdfhvfvrdv lqhvdsmqkd 121 ypglpvfllg hsmggaiail taaerpghfa gmvlisplvl anpesattfk vlaakvlnlv 181 lpnlslgpid ssvlsrnkte vdiynsdpli craglkvcfg iqllnavsrv eralpkltvp 241 flllqgsadr lcdskgayll melaksqdkt lkiyegayhv lhkelpevtn svfheinmwv 301 sqrtatagta spp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madssgqqgk grrvqpqwsp pagtqpcrlh lynsltrnke vfipqdgkkv twyccgptvy 61 dashmghars yisfdilrrv lkdyfkfdvf ycmnitdidd kiikrarqnh lfeqyrekrp 121 eaaqlledvq aalkpfsvkl nettdpdkkq mleriqhavq lateplekav qsrltgeevn 181 scvevlleea kdllsdwlds tlgcdvtdns ifsklpkfwe gdfhrdmeal nvlppdvltr 241 vseyvpeivn fvqkivdngy gyvsngsvyf dtakfassek hsygklvpea vgdqkalqeg 301 egdlsisadr lsekrspndf alwkaskpge pswpcpwgkg rpgwhiecsa magtllgasm 361 dihgggfdlr fphhdnelaq seayfendcw vryflhtghl tiagckmsks lknfitikda 421 lkkhsarqlr laflmhswkd tldyssntme salqyekfln efflnvkdil rapvditgqf 481 ekwgeeeael nknfydkkta ihkalcdnvd trtvmeemra lvsqcnlyma arkavrkrpn 541 qallenialy lthmlkifga veedsslgfp vggpgtslsl eatvmpylqv lsefregvrk 601 iareqkvpei lqlsdalrdn ilpelgvrfe dheglptvvk lvdrntllke reekrrveee 661 krkkkeeaar rkqeqeaakl akmkippsem flsetdkysk fdenvsmvcp hmtwrakssa 721 kgkprs';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count38 = 0; count38 < 4500; count38++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaldslslf tslglseqka retlknsals aqlreaatqa qqtlgstidk atgillygla 61 srlrdtrrls flvsyiaskk ihtepqlsaa leyvrshpld pidtvdfere cgvgvivtpe 121 qieeaveaai nrhrpqllve ryhfnmgllm gearavlkwa dgkmiknevd mqvlhllgpk 181 leadlekkfk vakarleetd rrtakdvven getadqtlsl meqlrgealk fhkpgenykt 241 pgyvvtphtm nllkqhleit ggqvrtrfpp epngilhigh akainfnfgy akanngicfl 301 rfddtnpeke eakfftaicd mvawlgytpy kvtyasdyfd qlyawaveli rrglayvchq 361 rgeelkghnt lpspwrdrpm eeslllfeam rkgkfsegea tlrmklvmed gkmdpvayrv 421 kytphhrtgd kwciyptydy thclcdsieh ithslctkef qarrssyfwl cnaldvycpv 481 qweygrlnlh yavvskrkil qlvatgavrd wddprlftlt alrrrgfppe ainnfcarvg 541 vtvaqttmep hlleacvrdv lndtaprama vleslrviit nfpaaksldi qvpnfpadet 601 kgfhqvpfap ivfiertdfk eepepgfkrl awgqpvglrh tgyvielqhv vkgpsgcves 661 levtcrrada gekpkafihw vsqplmcevr lyerlfqhkn pedptevpgg flsdlnlasl 721 hvvdaalvdc svalakpfdk fqferlgyfs vdpdshqgkl vfnrtvtlke dpgkv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matlsltvns gdpplgalla vehvkddvsi sveegkenil hvsenviftd vnsilrylar 61 vattaglygs nlmehteidh wlefsatkls scdsftstin elnhclslrt ylvgnslsla 121 dlcvwatlkg naawqeqlkq kkapvhvkrw fgfleaqqaf qsvgtkwdvs ttkarvapek 181 kqdvgkfvel pgaemgkvtv rfppeasgyl highakaall nqhyqvnfkg klimrfddtn 241 pekekedfek viledvamlh ikpdqftyts dhfetimkya ekliqegkay vddtpaeqmk 301 aereqridsk hrknpieknl qmweemkkgs qfgqscclra kidmssnngc mrdptlyrck 361 iqphprtgnk ynvyptydfa cpivdsiegv thalrtteyh drdeqfywii ealgirkpyi 421 weysrlnlnn tvlskrkltw fvneglvdgw ddprfptvrg vlrrgmtveg lkqfiaaqgs 481 srsvvnmewd kiwafnkkvi dpvapryval lkkevipvnv peaqeemkev akhpknpevg 541 lkpvwyspkv fiegadaetf segemvtfin wgnlnitkih knadgkiisl daklnlenkd 601 ykkttkvtwl aetthalpip vicvtyehli tkpvlgkded fkqyvnknsk heelmlgdpc 661 lkdlkkgdii qlqrrgffic dqpyepvspy sckeapcvli yipdghtkem ptsgskektk 721 veatknetsa pfkerptpsl nnncttseds lvlynrvavq gdvvrelkak kapkedvdaa 781 vkqllslkae ykektgqeyk pgnppaeigq nissnssasi leskslydev aaqgevvrkl 841 kaekspkaki neavecllsl kaqykektgk eyipgqppls qssdssptrn sepagletpe 901 akvlfdkvas qgevvrklkt ekapkdqvdi avqellqlka qyksligvey kpvsatgaed 961 kdkkkkeken ksekqnkpqk qndgqrkdps knqggglsss gagegqgpkk qtrlgleakk 1021 eenladwysq vitksemiey hdisgcyilr pwayaiweai kdffdaeikk lgvencyfpm 1081 fvsqsaleke kthvadfape vawvtrsgkt elaepiairp tsetvmypay akwvqshrdl 1141 piklnqwcnv vrwefkhpqp flrtreflwq eghsafatme eaaeevlqil dlyaqvyeel 1201 laipvvkgrk tekekfaggd ytttieafis asgraiqggt shhlgqnfsk mfeivfedpk 1261 ipgekqfayq nswglttrti gvmtmvhgdn mglvlpprva cvqvviipcg itnalseedk 1321 ealiakcndy rrrllsvnir vradlrdnys pgwkfnhwel kgvpirlevg prdmkscqfv 1381 avrrdtgekl tvaeneaetk lqailediqv tlftrasedl kthmvvantm edfqkildsg 1441 kivqipfcge idcedwikkt tardqdlepg apsmgakslc ipfkplcelq pgakcvcgkn 1501 pakyytlfgr sy';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count40 = 0; count40 < 4500; count40++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlqqvpenin fpaeeekile fwtefncfqe clkqskhkpk ftfydgppfa tglphyghil 61 agtikdivtr yahqsgfhvd rrfgwdchgl pveyeidktl girgpedvak mgiteynnqc 121 raivmrysae wkstvsrlgr widfdndykt lypqfmesvw wvfkqlydkg lvyrgvkvmp 181 fstacntpls nfeshqnykd vqdpsvfvtf pleedetvsl vawtttpwtl psnlavcvnp 241 emqyvkikdv argrllilme arlsalykle sdyeilerfp gaylkgkkyr plfdyflkck 301 engaftvlvd nyvkeeegtg vvhqapyfga edyrvcmdfn iirkdslpvc pvdasgcftt 361 evtdfagqyv kdadksiirt lkeqgrllva ttfthsypfc wrsdtpliyk avpswfvrve 421 nmvdqllrnn dlcywvpelv rekrfgnwlk dardwtisrn rywgtpiplw vsddfeevvc 481 igsvaeleel sgakisdlhr esvdhltips rcgkgslhri sevfdcwfes gsmpyaqvhy 541 pfenkrefed afpadfiaeg idqtrgwfyt llvlatalfg qppfknvivn glvlasdgqk 601 mskrkknypd pvsiiqkyga dalrlylins pvvraenlrf keegvrdvlk dvllpwynay 661 rfliqnvlrl qkeeeiefly nentvrespn itdrwilsfm qsligffete maayrlytvv 721 prlvkfvdil tnwyvrmnrr rlkgengmed cvmaletlfs vllslcrlma pytpfltelm 781 yqnlkvlidp vsvqdkdtls ihylmlprvr eelidkktes avsqmqsvie lgrvirdrkt 841 ipikyplkei vvihqdpeal kdikslekyi ieelnvrkvt lstdknkygi rlraepdhmv 901 lgkrlkgafk avmtsikqls seeleqfqkt gtivveghel hdedirlmyt fdqatggtaq 961 feahsdaqal vlldvtpdqs mvdegmarev inriqklrkk cnlvptdeit vyykaksegt 1021 ylnsviesht efifttikap lkpypvspsd kvliqektql kgseleitlt rgsslpgpac 1081 ayvnlnican gseqggvlll enpkgdnrld llklksvvts ifgvkntela vfhdeteiqn 1141 qtdllslsgk tlcvtagsap slinssstll cqyinlqlln akpqeclmgt vgtlllenpl 1201 gqnglthqgl lyeaakvfgl rsrklklfln etqtqeited ipvktlnmkt vyvsvlptta 1261 df';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maerkgtakv dflkkiekei qqkwdtervf evnasnlekq tskgkyfvtf pypymngrlh 61 lghtfslskc efavgyqrlk gkcclfpfgl hctgmpikac adklkreiel ygcppdfpde 121 eeeeeetsvk tediiikdka kgkkskaaak agsskyqwgi mkslglsdee ivkfseaehw 181 ldyfpplaiq dlkrmglkvd wrrsfittdv npyydsfvrw qfltlrernk ikfgkrytiy 241 spkdgqpcmd hdrqtgegvg pqeytllklk vlepypskls glkgkniflv aatlrpetmf 301 gqtncwvrpd mkyigfetvn gdifictqka arnmsyqgft kdngvvpvvk elmgeeilga 361 slsapltsyk viyvlpmlti kedkgtgvvt svpsdspddi aalrdlkkkq alrakygird 421 dmvlpfepvp vieipgfgnl savticdelk iqsqndrekl aeakekiylk gfyegimlvd 481 gfkgqkvqdv kktiqkkmid agdaliymep ekqvmsrssd ecvvalcdqw yldygeenwk 541 kqtsqclknl etfceetrrn featlgwlqe hacsrtyglg thlpwdeqwl ieslsdstiy 601 mafytvahll qggnlhgqae splgirpqqm tkevwdyvff keapfpktqi akekldqlkq 661 efefwypvdl rvsgkdlvpn hlsyylynhv amwpeqsdkw ptavranghl llnsekmsks 721 tgnfltltqa idkfsadgmr laladagdtv edanfveama dagilrlytw vewvkemvan 781 wdslrsgpas tfndrvfase lnagiiktdq nyekmmfkea lktgffefqa akdkyrelav 841 egmhrelvfr fievqtllla pfcphlcehi wtllgkpdsi mnaswpvagp vnevlihssq 901 ylmevthdlr lrlknymmpa kgkktdkqpl qkpshctiyv aknyppwqht tlsvlrkhfe 961 anngklpdnk viaselgsmp elkkymkkvm pfvamikenl ekmgprildl qlefdekavl 1021 menivyltns lelehievkf aseaedkire dccpgkplnv friepgvsvs lvnpqpsngh 1081 fstkieirqg dncdsiirrl mkmnrgikdl skvklmrfdd pllgprrvpv lgkeytektp 1141 isehavfnvd lmskkihlte ngirvdigdt iiylvh';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count42 = 0; count42 < 4500; count42++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrlfvsdgvp gclpvlaaag rargraevli stvgpedcvv pfltrpkvpv lqldsgnylf 61 stsaicryff llsgweqddl tnqwleweat elqpalsaal yylvvqgkkg edvlgsvrra 121 lthidhslsr qncpflaget esladivlwg alypllqdpa ylpeelsalh swfqtlstqe 181 pcqraaetvl kqqgvlalrp ylqkqpqpsp aegravtnep eeeelatlse eeiamavtaw 241 ekgleslppl rpqqnpvlpv agernvlits alpyvnnvph lgniigcvls advfarysrl 301 rqwntlylcg tdeygtatet kaleegltpq eicdkyhiih adiyrwfnis fdifgrtttp 361 qqtkitqdif qqllkrgfvl qdtveqlrce hcarfladrf vegvcpfcgy eeargdqcdk 421 cgklinavel kkpqckvcrs cpvvqssqhl fldlpklekr leewlgrtlp gsdwtpnaqf 481 itrswlrdgl kprcitrdlk wgtpvplegf edkvfyvwfd atigylsita nytdqwerww 541 knpeqvdlyq fmakdnvpfh slvfpcsalg aednytlvsh liateylnye dgkfsksrgv 601 gvfgdmaqdt gipadiwrfy llyirpegqd safswtdlll knnsellnnl gnfinragmf 661 vskffggyvp emvltpddqr llahvtlelq hyhqllekvr irdalrsilt isrhgnqyiq 721 vnepwkrikg seadrqragt vtglavniaa llsvmlqpym ptvsatiqaq lqlpppacsi 781 lltnflctlp aghqigtvsp lfqklendqi eslrqrfggg qaktspkpav vetvttakpq 841 qiqalmdevt kqgnivrelk aqkadkneva aevaklldlk kqlavaegkp peapkgkkkk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdvlvsecsa rllqqeeeik sltaeidrlk ncgclgaspn leqlqeenlk lkyrlnilrk 61 slqaernkpt knminiisrl qevfghaika aypdlenppl lvtpsqqakf gdyqcnsamg 121 isqmlktkeq kvnpreiaen itkhlpdnec iekveiagpg finvhlrkdf vseqltsllv 181 ngvqlpalge nkkvivdfss pniakemhvg hlrstiiges isrlfefagy dvlrlnhvgd 241 wgtqfgmlia hlqdkfpdyl tvsppigdlq vfykeskkrf dteeefkkra yqcvvllqgk 301 npditkawkl icdvsrqeln kiydaldvsl iergesfyqd rmndivkefe drgfvqvddg 361 rkivfvpgcs ipltivksdg gytydtsdla aikqrlfeek admiiyvvdn gqsvhfqtif 421 aaaqmigwyd pkvtrvfhag fgvvlgedkk kfktrsgetv rlmdllgegl krsmdklkek 481 erdkvltaee lnaaqtsvay gcikyadlsh nrlndyifsf dkmlddrgnt aayllyaftr 541 irsiarlani deemlqkaar etkilldhek ewklgrcilr fpeilqkild dlflhtlcdy 601 iyelatafte fydscycvek drqtgkilkv nmwrmllcea vaavmakgfd ilgikpvqrm';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count44 = 0; count44 < 4500; count44++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mstlyvsphp dafpslrali aarygeageg pgwggahpri clqppptsrt pfppprlpal 61 eqgpgglwvw gatavaqllw paglggpggs raavlvqqwv syadtelipa acgatlpalg 121 lrssaqdpqa vlgalgrals pleewlrlht ylageaptla dlaavtalll pfryvldppa 181 rriwnnvtrw fvtcvrqpef ravlgevvly sgarplshqp gpeapalpkt aaqlkkeakk 241 reklekfqqk qkiqqqqppp gekkpkpekr ekrdpgvity dlptppgekk dvsgpmpdsy 301 spryveaawy pwweqqgffk peygrpnvsa anprgvfmmc ipppnvtgsl hlghaltnai 361 qdsltrwhrm rgettlwnpg cdhagiatqv vvekklwreq glsrhqlgre aflqevwkwk 421 eekgdriyhq lkklgssldw dracftmdpk lsaavteafv rlheegiiyr strlvnwsct 481 lnsaisdiev dkkeltgrtl lsvpgykekv efgvlvsfay kvqgsdsdee vvvattriet 541 mlgdvavavh pkdtryqhlk gknvihpfls rslpivfdef vdmdfgtgav kitpahdqnd 601 yevgqrhgle aisimdsrga linvpppflg lprfearkav lvalkerglf rgiednpmvv 661 plcnrskdvv epllrpqwyv rcgemaqaas aavtrgdlri lpeahqrtwh awmdnirewc 721 isrqlwwghr ipayfvtvsd pavppgedpd grywvsgrne aearekaake fgvspdkisl 781 qqdedvldtw fssglfplsi lgwpnqsedl svfypgtlle tghdilffwv armvmlglkl 841 tgrlpfrevy lhaivrdahg rkmskslgnv idpldviygi slqglhnqll nsnldpseve 901 kakegqkadf pagipecgtd alrfglcaym sqgrdinldv nrilgyrhfc nklwnatkfa 961 lrglgkgfvp sptsqpgghe slvdrwirsr lteavrlsnq gfqaydfpav ttaqysfwly 1021 elcdvylecl kpvlngvdqv aaecarqtly tcldvglrll spfmpfvtee lfqrlprrmp 1081 qappslcvtp ypepsecswk dpeaeaalel alsitravrs lradynltri rpdcflevad 1141 eatgalasav sgyvqalasa gvvavlalga papqgcaval asdrcsihlq lqglvdpare 1201 lgklqakrve aqrqaqrlre rraasgypvk vplevqeade aklqqteael rkvdeaialf 1261 qkml';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpnsepasll elfnsiatqg elvrslkagn askdeidsav kmlvslkmsy kaaagedyka 61 dcppgnpapt snhgpdatea eedfvdpwtv qtssakgidy dklivrfgss kidkelinri 121 eratgqrphh flrrgiffsh rdmnqvlday enkkpfylyt grgpsseamh vghlipfift 181 kwlqdvfnvp lviqmtddek ylwkdltldq aysyavenak diiacgfdin ktfifsdldy 241 mgmssgfykn vvkiqkhvtf nqvkgifgft dsdcigkisf paiqaapsfs nsfpqifrdr 301 tdiqclipca idqdpyfrmt rdvaprigyp kpallhstff palqgaqtkm sasdpnssif 361 ltdtakqikt kvnkhafsgg rdtieehrqf ggncdvdvsf myltffledd dkleqirkdy 421 tsgamltgel kkalievlqp liaehqarrk evtdeivkef mtprklsfdf q';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count46 = 0; count46 < 4500; count46++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgdapspeek lhlitrnlqe vlgeeklkei lkerelkiyw gtattgkphv ayfvpmskia 61 dflkagcevt ilfadlhayl dnmkapwell elrvsyyenv ikamlesigv pleklkfikg 121 tdyqlskeyt ldvyrlssvv tqhdskkaga evvkqvehpl lsgllypglq aldeeylkvd 181 aqfggidqrk iftfaekylp algyskrvhl mnpmvpgltg skmssseees kidlldrked 241 vkkklkkafc epgnvenngv lsfikhvlfp lksefvilrd ekwggnktyt ayvdlekdfa 301 aevvhpgdlk nsvevalnkl ldpirekfnt palkklasaa ypdpskqkpm akgpaknsep 361 eevipsrldi rvgkiitvek hpdadslyve kidvgeaepr tvvsglvqfv pkeelqdrlv 421 vvlcnlkpqk mrgvesqgml lcasieginr qvepldppag sapgehvfvk gyekgqpdee 481 lkpkkkvfek lqadfkisee ciaqwkqtnf mtklgsisck slkggnis';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdstltasei rqrfidffkr nehtyvhssa tiplddptll fanagmnqfk piflntidps 61 hpmaklsraa ntqkciragg khndlddvgk dvyhhtffem lgswsfgdyf kelackmale 121 lltqefgipi erlyvtyfgg deaagleadl eckqiwqnlg lddtkilpgn mkdnfwemgd 181 tgpcgpcsei hydriggrda ahlvnqddpn vleiwnlvfi qynreadgil kplpkksidt 241 gmglerlvsv lqnkmsnydt dlfvpyfeai qkgtgarpyt gkvgaedadg idmayrvlad 301 hartitvala dggrpdntgr gyvlrrilrr avryahekln asrgffatlv dvvvqslgda 361 fpelkkdpdm vkdiineeev qflktlsrgr rildrkiqsl gdsktipgdt awllydtygf 421 pvdltgliae ekglvvdmdg feeerklaql ksqgkgagge dlimldiyai eelrarglev 481 tddspkynyh ldssgsyvfe ntvatvmalr rekmfveevs tgqecgvvld ktcfyaeqgg 541 qiydegylvk vddssedkte ftvknaqvrg gyvlhigtiy gdlkvgdqvw lfideprrrp 601 imsnhtathi lnfalrsvlg eadqkgslva pdrlrfdfta kgamstqqik kaeeianemi 661 eaakavytqd cplaaakaiq glravfdety pdpvrvvsig vpvsellddp sgpagsltsv 721 efcggthlrn sshagafviv teeaiakgir rivavtgaea qkalrkaesl kkclsvmeak 781 vkaqtapnkd vqreiadlge alatavipqw qkdelretlk slkkvmddld raskadvqkr 841 vlektkqfid snpnqplvil emesgasaka lnealklfkm hspqtsamlf tvdneagkit 901 clcqvpqnaa nrglkasewv qqvsglmdgk gggkdvsaqa tgknvgclqe alqlatsfaq 961 lrlgdvkn';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count48 = 0; count48 < 4500; count48++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpsasasrks qekpreimda aedyakeryg issmiqsqek pdrvlvrvrd ltiqkadevv 61 wvrarvhtsr akgkqcflvl rqqqfnvqal vavgdhaskq mvkfaanink esivdvegvv 121 rkvnqkigsc tqqdvelhvq kiyvislaep rlplqlddav rpeaegeeeg ratvnqdtrl 181 dnrvidlrts tsqavfrlqs gichlfretl inkgfveiqt pkiisaaseg ganvftvsyf 241 knnaylaqsp qlykqmcica dfekvfsigp vfraedsnth rhltefvgld iemafnyhyh 301 evmeeiadtm vqifkglqer fqteiqtvnk qfpcepfkfl eptlrleyce alamlreagv 361 emgdeddlst pnekllghlv kekydtdfyi ldkyplavrp fytmpdprnp kqsnsydmfm 421 rgeeilsgaq rihdpqllte ralhhgidle kikayidsfr fgapphaggg iglervtmlf 481 lglhnvrqts mfprdpkrlt p';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madgqvaell lrrleasdgg ldsaelaael gmehqavvga vkslqalgev ieaelrstkh 61 weltaegeei aregshearv frsippegla qselmrlpsg kvgfskamsn kwirvdksaa 121 dgprvfrvvd smedevqrrl qlvrggqaek lgekerselr krkllaevtl ktywvskgsa 181 fstsiskqet elspemissg swrdrpfkpy nflahgvlpd sghlhpllkv rsqfrqifle 241 mgftemptdn fiessfwnfd alfqpqqhpa rdqhdtfflr dpaealqlpm dyvqrvkrth 301 sqggygsqgy kynwkldear knllrthtts asaralyrla qkkpftpvky fsidrvfrne 361 tldathlaef hqiegvvadh gltlghlmgv lrefftklgi tqlrfkpayn pytepsmevf 421 syhqglkkwv evgnsgvfrp emllpmglpe nvsviawgls lerptmikyg innirelvgh 481 kvnlqmvyds plcrldaepr ppptqeaa';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count50 = 0; count50 < 4500; count50++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mptvsvkrdl lfqalgrtyt deefdelcfe fgleldeits ekeiiskeqg nvkaagasdv 61 vlykidvpan rydllclegl vrglqvfker ikapvykrvm pdgkiqklii teetakirpf 121 avaavlrnik ftkdrydsfi elqeklhqni crkralvaig thdldtlsgp ftytakrpsd 181 ikfkplnktk eytacelmni yktdnhlkhy lhiienkply pviydsngvv lsmppiingd 241 hsritvntrn ifiectgtdf tkakivldii vtmfseycen qftveaaevv fpngkshtfp 301 elayrkemvr adlinkkvgi retpenlakl ltrmylksev igdgnqieie ipptradiih 361 acdivedaai aygynniqmt lpktytianq fplnkltell rhdmaaagft ealtfalcsq 421 ediadklgvd isatkavhis npktaefqva rttllpgllk tiaanrkmpl plklfeisdi 481 vikdsntdvg aknyrhlcav yynknpgfei ihglldrimq lldvppgedk ggyvikaseg 541 paffpgrcae ifargqsvgk lgvlhpdvit kfeltmpcss leinvgpfl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpsprpvllr garaalllll pprllarpsl llrrslsaas cppislpaaa srssmdgaga 61 eevlaplrla vrqqgdlvrk lkedkapqvd vdkavaelka rkrvleakel alqpkddivd 121 rakmedtlkr rffydqafai yggvsglydf gpvgcalknn iiqtwrqhfi qeeqileidc 181 tmltpepvlk tsghvdkfad fmvkdvknge cfradhllka hlqklmsdkk csvekkseme 241 svlaqldnyg qqeladlfvn ynvkspitgn dlsppvsfnl mfktfigpgg nmpgylrpet 301 aqgiflnfkr llefnqgklp faaaqignsf rneisprsgl irvreftmae iehfvdpsek 361 dhpkfqnvad lhlylysaka qvsgqsarkm rlgdaveqgv inntvlgyfi griylyltkv 421 gispdklrfr qhmenemahy acdcwdaesk tsygwieivg cadrscydls charatkvpl 481 vaekplkepk tvnvvqfeps kgaigkaykk daklvmeyla icdecyitem emllnekgef 541 tietegktfq ltkdminvkr fqktlyveev vpnviepsfg lgrimytvfe htfhvregde 601 qrtffsfpav vapfkcsvlp lsqnqefmpf vkelsealtr hgvshkvdds sgsigrryar 661 tdeigvafgv tidfdtvnkt phtatlrdrd smrqiraeis elpsivqdla ngnitwadve 721 aryplfegqe tgkketiee';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count52 = 0; count52 < 4500; count52++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdgagaeevl aplrlavrqq gdlvrklked kapqvdvdka vaelkarkrv leakelalqp 61 kddivdrakm edtlkrrffy dqafaiyggv sglydfgpvg calknniiqt wrqhfiqeeq 121 ileidctmlt pepvlktsgh vdkfadfmvk dvkngecfra dhllkahlqk lmsdkkcsve 181 kksemesvla qldnygqqel adlfvnynvk spitgndlsp pvsfnlmfkt figpggnmpg 241 ylrpetaqgi flnfkrllef nqgklpfaaa qignsfrnei sprsglirvr eftmaeiehf 301 vdpsekdhpk fqnvadlhly lysakaqvsg qsarkmrlgd aveqgvinnt vlgyfigriy 361 lyltkvgisp dklrfrqhme nemahyacdc wdaesktsyg wieivgcadr scydlschar 421 atkvplvaek plkepktvnv vqfepskgai gkaykkdakl vmeylaicde cyitememll 481 nekgeftiet egktfqltkd minvkrfqkt lyveevvpnv iepsfglgri mytvfehtfh 541 vregdeqrtf fsfpavvapf kcsvlplsqn qefmpfvkel sealtrhgvs hkvddssgsi 601 grryartdei gvafgvtidf dtvnktphta tlrdrdsmrq iraeiselps ivqdlangni 661 twadvearyp lfegqetgkk etiee';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeraaleel vklqgervrg lkqqkasael ieeevakllk lkaqlgpdes kqkfvlktpk 61 gtrdysprqm avrekvfdvi ircfkrhgae vidtpvfelk etlmgkyged skliydlkdq 121 ggellslryd ltvpfaryla mnkltnikry hiakvyrrdn pamtrgryre fyqcdfdiag 181 nfdpmipdae clkimceils slqigdflvk vndrrildgm faicgvsdsk frticssvdk 241 ldkvsweevk nemvgekgla pevadrigdy vqqhggvslv eqllqdpkls qnkqaleglg 301 dlkllfeylt lfgiddkisf dlslargldy ytgviyeavl lqtpaqagee plgvgsvaag 361 grydglvgmf dpkgrkvpcv glsigverif siveqrleal eekirttetq vlvasaqkkl 421 leerlklvse lwdagikael lykknpklln qlqyceeagi plvaiigeqe lkdgviklrs 481 vtsreevdvr redlveeikr rtgqplcic';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count54 = 0; count54 < 4500; count54++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvlaelyvsd regsdatgdg tkekpfktgl kalmtvgkep fptiyvdsqk enerwnvisk 61 sqlknikkmw hreqmksesr ekkeaedslr reknleeakk itikndpslp epkcvkigal 121 egyrgqrvkv fgwvhrlrrq gknlmflvlr dgtgylqcvl adelcqcyng vllstessva 181 vygmlnltpk gkqapgghel scdfweligl apaggadnli neesdvdvql nnrhmmirge 241 nmskilkars mvtrcfrdhf fdrgyyevtp ptlvqtqveg gatlfkldyf geeafltqss 301 qlyletclpa lgdvfciaqs yraeqsrtrr hlaeythvea ecpfltfddl lnrledlvcd 361 vvdrilkspa gsivhelnpn fqppkrpfkr mnysdaivwl kehdvkkedg tfyefgedip 421 eaperlmtdt inepillcrf pveiksfymq rcpedsrlte svdvlmpnvg eivggsmrif 481 dseeilagyk regidptpyy wytdqrkygt cphggyglgl erfltwilnr yhirdvclyp 541 rfvqrctp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mltqaavrlv rgslrktswa ewghrelrlg qlapftaphk dksfsdqrse lkrrlkaekk 61 vaekeakqke lsekqlsqat aaatnhttdn gvgpeeesvd pnqyykirsq aihqlkvnge 121 dpyphkfhvd isltdfiqky shlqpgdhlt ditlkvagri hakrasggkl ifydlrgegv 181 klqvmansrn ykseeefihi nnklrrgdii gvqgnpgktk kgelsiipye itllspclhm 241 lphlhfglkd ketryrqryl dlilndfvrq kfiirskiit yirsfldelg fleietpmmn 301 iipggavakp fityhneldm nlymriapel yhkmlvvggi drvyeigrqf rnegidlthn 361 pefttcefym ayadyhdlme itekmvsgmv khitgsykvt yhpdgpegqa ydvdftppfr 421 rinmveelek algmklpetn lfeteetrki lddicvakav ecppprttar lldklvgefl 481 evtcinptfi cdhpqimspl akwhrskegl terfelfvmk keicnaytel ndpmrqrqlf 541 eeqakakaag ddeamfiden fctaleyglp ptagwgmgid rvamfltdsn nikevllfpa 601 mkpedkkenv attdtlestt vgtsv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count56 = 0; count56 < 4500; count56++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvldldlfrv dkggdpalir etqekrfkdp glvdqlvkad sewrrcrfra dnlnklknlc 61 sktigekmkk kepvgddesv penvlsfddl tadalanlkv sqikkvrlli deailkcdae 121 rikleaerfe nlreignllh psvpisnded vdnkveriwg dctvrkkysh vdlvvmvdgf 181 egekgavvag srgyflkgvl vfleqaliqy alrtlgsrgy ipiytpffmr kevmqevaql 241 sqfdeelykv igkgseksdd nsydekylia tseqpiaalh rdewlrpedl pikyaglstc 301 frqevgshgr dtrgifrvhq fekieqfvys sphdnkswem feemittaee fyqslgipyh 361 ivnivsgsln haaskkldle awfpgsgafr elvscsnctd yqarrlriry gqtkkmmdkv 421 efvhmlnatm cattrticai lenyqtekgi tvpekfmppg lqelipfvkp apieqepskk 481 qkkqhegskk kaaardvtle nrlqnmevtd a';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfeekassps gkmggeekpi gageekqkeg gkkknkegsg dggraelnpw peyiytrlem 61 ynilkaehds ilaekaekds kpikvtlpdg kqvdaeswkt tpyqiacgis qgladntvia 121 kvnnvvwdld rpleedctle llkfedeeaq avywhssahi mgeamervyg gclcygppie 181 ngfyydmyle eggvssndfs slealckkii kekqaferle vkketllamf kynkfkcril 241 nekvntpttt vyrcgplidl crgphvrhtg kikalkihkn sstywegkad metlqriygi 301 sfpdpkmlke wekfqeeakn rdhrkigrdq elyffhelsp gscfflpkga yiynaliefi 361 rseyrkrgfq evvtpnifns rlwmtsghwq hysenmfsfe vekelfalkp mncpghclmf 421 dhrprswrel plrladfgvl hrnelsgalt gltrvrrfqq ddahifcame qiedeikgcl 481 dflrtvysvf gfsfklnlst rpekflgdie vwdqaekqle nslnefgekw elnsgdgafy 541 gpkidiqikd aigryhqcat iqldfqlpir fnltyvshdg ddkkrpvivh railgsverm 601 iailtenygg kwpfwlsprq vmvvpvgptc deyaqkvrqq fhdakfmadi dldpgctlnk 661 kirnaqlaqy nfilvvgeke kisgtvnirt rdnkvhgert isetierlqq lkefrskqae 721 eef';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count58 = 0; count58 < 4500; count58++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 madskegvlp ltaastapis fgftrtsarr rladsgdgag pspeekdflk tvegrelqsv 61 kpqeapkelv ipliqnghrr qpparppgps tdtgaladgv vsqavkelia eskksleere 121 nagvdptlai pmiqkgctps gegadsepra etvpeeanye avpveaygla mlrgmgwkpg 181 egigrtfnqv vkprvnslrp kglglganlt eaqaltptgp srmprpdeeq ekdkedqpqg 241 lvpggavvvl sgphrglygk vegldpdnvr amvrlavgsr vvtvseyylr pvsqqefdkn 301 tldlrqqngt assrktlwnq elyiqqdnse rkrkhlpdrq dgpaakseka aprsqhwlhr 361 dlrvrfvdnm ykggqyyntk miiedvlspd tcvcrtdegr vleglredml etlvpkaegd 421 rvmvvlgpqt grvghllsrd rarsralvql prenqvvelh ydaicqymgp sdtddd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrrstsstk sgkfmnptdq arkearkrel kknkkqrmmv raavlkmkdp kqiirdmekl 61 demefnpvqq pqlnekvlkd krkklretfe rilrlyeken pdiykelrkl eveyeqkraq 121 lsqyfdavkn aqhvevesip lpdmphapsn iliqdiplpg aqppsilkkt saygpptrav 181 silpllghgv prlppgrkpp gpppgppppq vvqmygrkvg faldlpprrr dedmlyspel 241 aqrghdddvs stseddgype dmdqdkhdds tddsdtdksd gesdgdefvh rdngerdnne 301 ekksglsvrf admpgksrkk kknmkeltpl qammlrmagq eipeegreve efsedddedd 361 sddseaekqs qkqhkeeshs dgtstassqq qappqsvpps qiqappmpgp pplgpppapp 421 lrppgpptgl ppgpppgapp flrppgmpgl rgplprllpp gpppgrppgp ppgpppglpp 481 gppprgpppr lpppappgip pprpgmmrpp lvpplgpapp glfppaplpn pgvlsappnl 541 iqrpkaddts aatiekkata tisakpqitn pkaeitrfvp talrvrrenk gataapqrks 601 eddsavplak aapksgpsvp vsvqtkddvy eafmkemegl l';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count60 = 0; count60 < 4500; count60++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madskegvlp ltaastapis fgftrtsarr rladsgdgag pspeekdflk tvegrelqsv 61 kpqeapkelv ipliqnghrr qpparppgps tdtgaladgv vsqavkelia eskksleere 121 nagvdptlai pmiqkgctps gegadsepra etvpeeanye avpveaygla mlrgmgwkpg 181 egigrtfnqv vkprvnslrp kglglganlt eaqaltptgp srmprpdeeq ekdkedqpqg 241 lvpggavvvl sgphrglygk vegldpdnvr amvrlavgsr vvtvseyylr pvsqqefdkn 301 tldlrqqngt assrktlwnq elyiqqdnse rkrkhlpdrq dgpaakseka aprsqhwlhr 361 dlrvrfvdnm ykggqyyntk miiedvlspd tcvcrtdegr vleglredml etlvpkaegd 421 rvmvvlgpqt grvghllsrd rarsralvql prenqvvelh ydaicqymgp sdtddd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgkrqhqkdk myitcaeyth fyggkkpdlp qtnfrrlpfd hcslslqpfv ypvctpdgiv 61 fdllnivpwl kkygtnpsng ekldgrslik lnfsknsegk yhcpvlftvf tnnthivavr 121 ttgnvyayea veqlnikakn frdlltdepf srqdiitlqd ptnldkfnvs nfyhvknnmk 181 iidpdeekak qdpsyylknt naetretlqe lykefkgdei laatmkapek kkvdklnaah 241 ystgkvsasf tstamvpett heaaaidedv lryqfvkkkg yvrlhtnkgd lnlelhcdlt 301 pktcenfirl ckkhyydgti fhrsirnfvi qggdptgtgt ggesywgkpf kdefrpnlsh 361 tgrgilsman sgpnsnrsqf fitfrscayl dkkhtifgrv vggfdvltam envesdpktd 421 rpkeeirida ttvfvdpyee adaqiaqerk tqlkvapetk vkssqpqags qgpqtfrqgv 481 gkyinpaatk raaeeepsts atvpmskkkp srgfgdfssw';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count62 = 0; count62 < 4500; count62++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'mqifvktltg ktitleveps dtienvkaki qdkegippdq qrlifagkql edgrtlsdyn 61 iqkestlhlv lrlrgg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mslvayassd esepdeaepe peeeeavapt sgpalgglfa slpapkgpal lppppqmlap 61 afppplllpp ptgdprlqpp pplpfglggf ppppgvspae aagvgeglgl glpsprgpgl 121 nlpppiggag pplglpkpkk rkepvkiaap elhkgdsdse edeptkkkti lqgssegtgl 181 sallpqpknl tvketnrlll phafsrkpsd gspdtkpsrl asktktssla pvvgtttttp 241 spsaikaaak saalqvtkqi tqeeddsdee vapenffslp ekaeppgvep ypypiptvpe 301 elppgtepep afqddaanap lefkmaagss gapwmpkpgd dysynqfsty gdanaagayy 361 qdyysggyyp aqdpalvppq eiapdasfid deafkrlqgk rnrgreeinf veikgddqls 421 gaqqwmtksl teektmksfs kkkgeqptgq qrrkhqityl ihqakerele lkntwsenkl 481 srrqtqakyg f';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count64 = 0; count64 < 4500; count64++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 magvfpyrgp gnpvpgplap lpdymseekl qekarkwqql qakryaekrk fgfvdaqked 61 mppehvrkii rdhgdmtnrk frhdkrvylg alkymphavl kllenmpmpw eqirdvpvly 121 hitgaisfvn eipwviepvy isqwgsmwim mrrekrdrrh fkrmrfppfd deeppldyad 181 nildveplea iqleldpeed apvldwfydh qplrdsrkyv ngstyqrwqf tlpmmstlyr 241 lanqlltdlv ddnyfylfdl kafftskaln maipggpkfe plvrdinlqd edwnefndin 301 kiiirqpirt eykiafpyly nnlphhvhlt wyhtpnvvfi ktedpdlpaf yfdplinpis 361 hrhsvksqep lpdddeefel pefvepflkd tplytdntan giallwaprp fnlrsgrtrr 421 aldiplvknw yrehcpagqp vkvrvsyqkl lkyyvlnalk hrppkaqkkr ylfrsfkatk 481 ffqstkldwv evglqvcrqg ynmlnllihr knlnylhldy nfnlkpvktl ttkerkksrf 541 gnafhlcrev lrltklvvds hvqyrlgnvd afqladglqy ifahvgqltg myrykyklmr 601 qirmckdlkh liyyrfntgp vgkgpgcgfw aagwrvwlff mrgitpller wlgnllarqf 661 egrhskgvak tvtkqrvesh fdlelraavm hdildmmpeg ikqnkartil qhlseawrcw 721 kanipwkvpg lptpienmil ryvkakadww tntahynrer irrgatvdkt vckknlgrlt 781 rlylkaeqer qhnylkdgpy itaeeavavy tttvhwlesr rfspipfppl sykhdtklli 841 lalerlkeay svksrlnqsq reelglieqa ydnphealsr ikrhlltqra fkevgiefmd 901 lyshlvpvyd veplekitda yldqylwyea dkrrlfppwi kpadtepppl lvykwcqgin 961 nlqdvwetse gecnvmlesr fekmyekidl tllnrllrli vdhniadymt aknnvvinyk 1021 dmnhtnsygi irglqfasfi vqyyglvmdl lvlglhrase magppqmpnd flsfqdiate 1081 aahpirlfcr yidrihiffr ftadeardli qryltehpdp nnenivgynn kkcwprdarm 1141 rlmkhdvnlg ravfwdiknr lprsvttvqw ensfvsvysk dnpnllfnmc gfecrilpkc 1201 rtsyeefthk dgvwnlqnev tkertaqcfl rvddesmqrf hnrvrqilma sgsttftkiv 1261 nkwntaligl mtyfreavvn tqelldllvk cenkiqtrik iglnskmpsr fppvvfytpk 1321 elgglgmlsm ghvlipqsdl rwskqtdvgi thfrsgmshe edqlipnlyr yiqpwesefi 1381 dsqrvwaeya lkrqeaiaqn rrltledled swdrgiprin tlfqkdrhtl aydkgwrvrt 1441 dfkqyqvlkq npfwwthqrh dgklwnlnny rtdmiqalgg vegilehtlf kgtyfptweg 1501 lfwekasgfe esmkwkkltn aqrsglnqip nrrftlwwsp tinranvyvg fqvqldltgi 1561 fmhgkiptlk isliqifrah lwqkihesiv mdlcqvfdqe ldaleietvq ketihprksy 1621 kmnsscadil lfasykwnvs rpslladskd vmdstttqky widiqlrwgd ydshdierya 1681 rakfldyttd nmsiypsptg vliaidlayn lhsaygnwfp gskpliqqam akimkanpal 1741 yvlrerirkg lqlyssepte pylssqnyge lfsnqiiwfv ddtnvyrvti hktfegnltt 1801 kpingaifif nprtgqlflk iihtsvwagq krlgqlakwk taeevaalir slpveeqpkq 1861 iivtrkgmld plevhlldfp nivikgselq lpfqaclkve kfgdlilkat epqmvlfnly 1921 ddwlktissy tafsrlilil ralhvnndra kvilkpdktt itephhiwpt ltdeewikve 1981 vqlkdlilad ygkknnvnva sltqseirdi ilgmeisaps qqrqqiaeie kqtkeqsqlt 2041 atqtrtvnkh gdeiitstts nyetqtfssk tewrvraisa anlhlrtnhi yvssddiket 2101 gytyilpknv lkkficisdl raqiagylyg vsppdnpqvk eircivmvpq wgthqtvhlp 2161 gqlpqheylk emeplgwiht qpnespqlsp qdvtthakim adnpswdgek tiiitcsftp 2221 gsctltaykl tpsgyewgrq ntdkgnnpkg ylpshyervq mllsdrflgf fmvpaqsswn 2281 ynfmgvrhdp nmkyelqlan pkefyhevhr pshflnfall qegevysadr edlya';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 matslgsnty nrqnwedadf pilcqtclge npyirmtkek ygkeckicar pftvfrwcpg 61 vrmrfkktev cqtcsklknv cqtclldley glpiqvrdag lsfkddmpks dvnkeyytqn 121 mereisnsdg trpvgmlgka tstsdmllkl arttpyykrn rphicsfwvk geckrgeecp 181 yrhekptdpd dpladqnikd ryygindpva dkllkrastm prldppedkt ittlyvgglg 241 dtitetdlrn hfyqfgeirt itvvqrqqca fiqfatrqaa evaaeksfnk livngrrlnv 301 kwgrsqaarg kekekdgttd sgiklepvpg lpgalppppa aeeeasanyf nlppsgppav 361 vnialppppg iapppppgfg phmfhpmgpp ppfmrapgpi hypsqdpqrm gahagkhssp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count66 = 0; count66 < 4500; count66++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhllkvgtwr nntasswlmk fsvlwlvsqn ccrasvvwma ymnisfhvgn hvlselgetg 61 vfgrsstlkr vagvivppeg kiqnacnpnt ifsrskyset wlalierggc tftqkikvat 121 ekgasgviiy nvpgtgnqvf pmfhqafedv vvvmignlkg teifhlikkg vlitavvevg 181 rkhiiwmnhy lvsfvivtta tlayfifyhi hrlclariqn rrwqrlttdl qntfgqlqlr 241 vvkegdeein pngdscvicf erykpndivr iltckhffhk ncidpwilph gtcpickcdi 301 lkvlgiqvvv engteplqvl msnelpetls pseeetnnev spagtsdkvi hveenptsqn 361 ndiqphsvve dvhpsp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mlaagvggqg erlagrrrkm avesrvtqee ikkepekpid rektcplllr vfttnngrhh 61 rmdefsrgnv psselqiytw mdatlkelts lvkevypear kkgthfnfai vftdvkrpgy 121 rvkeigstms grkgtddsmt lqsqkfqigd yldiaitppn rapppsgrmr py';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count68 = 0; count68 < 4500; count68++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 madskegvlp ltaastapis fgftrtsarr rladsgdgag pspeekdflk tvegrelqsv 61 kpqeapkelv ipliqnghrr qpparppgps tdtgaladgv vsqavkelia eskksleere 121 nagvdptlai pmiqkgctps gegadsepra etvpeeanye avpveaygla mlrgmgwkpg 181 egigrtfnqv vkprvnslrp kglglganlt eaqaltptgp srmprpdeeq ekdkedqpqg 241 lvpggavvvl sgphrglygk vegldpdnvr amvrlavgsr vvtvseyylr pvsqqefdkn 301 tldlrqqngt assrktlwnq elyiqqdnse rkrkhlpdrq dgpaakseka aprsqhwlhr 361 dlrvrfvdnm ykggqyyntk miiedvlspd tcvcrtdegr vleglredml etlvpkaegd 421 rvmvvlgpqt grvghllsrd rarsralvql prenqvvelh ydaicqymgp sdtddd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mieqqkrkgp elplvpvkrq rhelllgags gpgagqqqat pgallqagpp rcsslqapim 61 llsghegevy cckfhpngst lasagfdrli llwnvygdcd nyatlkghsg avmelhyntd 121 gsmlfsastd ktvavwdset gervkrlkgh tsfvnscypa rrgpqlvctg sddgtvklwd 181 irkkaaiqtf qntyqvlavt fndtsdqiis ggidndikvw dlrqnkltyt mrghadsvtg 241 lslssegsyl lsnamdntvr vwdvrpfapk ercvkifqgn vhnfeknllr cswspdgski 301 aagsadrfvy vwdttsrril yklpghagsi nevafhpdep iiisassdkr lymgeiq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count70 = 0; count70 < 4500; count70++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madvtarslq yeykansnlv lqadrslidr trrdeptgev lslvgklegt rmgdkaqrtk 61 pqmqeerrak rrkrdedrhd inkmkgytll segidemvgi iykpktketr etyevllsfi 121 qaalgdqprd ilcgaadevl avlkneklrd kerrkeidll lgqtddtryh vlvnlgkkit 181 dyggdkeiqn mddnidetyg vnvqfesdee egdedvygev reeasdddme gdeavvrctl 241 sanlvasgel msskkkdlhp rdidafwlqr qlsrfyddai vsqkkadevl eilktasddr 301 ecenqlvlll gfntfdfikv lrqhrmmily ctllasaqse aekerimgkm eadpelskfl 361 yqlheteked lireersrre rvrqsrmdtd letmdldqgg ealaprqvld ledlvftqgs 421 hfmankrcql pdgsfrrqrk gyeevhvpal kpkpfgseeq llpveklpky aqagfegfkt 481 lnriqsklyr aaletdenll lcaptgagkt nvalmcmlre igkhinmdgt invddfkiiy 541 iapmrslvqe mvgsfgkrla tygitvaelt gdhqlckeei satqiivctp ekwdiitrkg 601 gertytqlvr liildeihll hddrgpvlea lvarairnie mtqedvrlig lsatlpnyed 661 vatflrvdpa kglfyfdnsf rpvpleqtyv gitekkaikr fqimneivye kimehagknq 721 vlvfvhsrke tgktaraird mclekdtlgl flregsaste vlrteaeqck nlelkdllpy 781 gfaihhagmt rvdrtlvedl fadkhiqvlv statlawgvn lpahtviikg tqvyspekgr 841 wtelgaldil qmlgragrpq ydtkgegili tshgelqyyl sllnqqlpie sqmvsklpdm 901 lnaeivlgnv qnakdavnwl gyaylyirml rsptlygish ddlkgdplld qrrldlvhta 961 almldknnlv kydkktgnfq vtelgriash yyitndtvqt ynqllkptls eielfrvfsl 1021 ssefknitvr eeeklelqkl lervpipvke sieepsakin vllqafisql klegfalmad 1081 mvyvtqsagr lmraifeivl nrgwaqltdk tlnlckmidk rmwqsmcplr qfrklpeevv 1141 kkiekknfpf erlydlnhne igelirmpkm gktihkyvhl fpklelsvhl qpitrstlkv 1201 eltitpdfqw dekvhgssea fwilvedvds evilhheyfl lkakyaqdeh litffvpvfe 1261 plppqyfirv vsdrwlscet qlpvsfrhli lpekypppte lldlqplpvs alrnsafesl 1321 yqdkfpffnp iqtqvfntvy nsddnvfvga ptgsgktica efailrmllq ssegrcvyit 1381 pmealaeqvy mdwyekfqdr lnkkvvlltg etstdlkllg kgniiistpe kwdilsrrwk 1441 qrknvqninl fvvdevhlig gengpvlevi csrmryissq ierpirival ssslsnakdv 1501 ahwlgcsats tfnfhpnvrp vplelhiqgf nishtqtrll smakpvyhai tkhspkkpvi 1561 vfvpsrkqtr ltaidilttc aadiqrqrfl hctekdlipy leklsdstlk etllngvgyl 1621 heglspmerr lveqlfssga iqvvvasrsl cwgmnvaahl viimdtqyyn gkihayvdyp 1681 iydvlqmvgh anrplqddeg rcvimcqgsk kdffkkflye plpveshldh cmhdhfnaei 1741 vtktienkqd avdyltwtfl yrrmtqnpny ynlqgishrh lsdhlselve qtlsdleqsk 1801 cisiedemdv aplnlgmiaa yyyinyttie lfsmslnakt kvrglieiis naaeyenipi 1861 rhhednllrq laqkvphkln npkfndphvk tnlllqahls rmqlsaelqs dteeilskai 1921 rliqacvdvl ssngwlspal aamelaqmvt qamwskdsyl kqlphftseh ikrctdkgve 1981 svfdimemed eernallqlt dsqiadvarf cnrypniels yevvdkdsir sggpvvvlvq 2041 lereeevtgp viaplfpqkr eegwwvvigd aksnslisik rltlqqkakv kldfvapatg 2101 ahnytlyfms daymgcdqey kfsvdvkeae tdsdsd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvvmarlsrp erpdlvfeee dlpyeeeimr nqfsvkcwlr yiefkqgapk prlnqlyera 61 lkllpcsykl wyrylkarra qvkhrcvtdp ayedvnnche rafvfmhkmp rlwldycqfl 121 mdqgrvthtr rtfdralral pitqhsriwp lylrflrshp lpetavrgyr rflklspesa 181 eeyieylkss drldeaaqrl atvvnderfv skagksnyql whelcdlisq npdkvqslnv 241 daiirggltr ftdqlgklwc sladyyirsg hfekardvye eairtvmtvr dftqvfdsya 301 qfeesmiaak metaselgre eeddvdlelr larfeqlisr rplllnsvll rqnphhvhew 361 hkrvalhqgr preiintyte avqtvdpfka tgkphtlwva fakfyedngq lddarvilek 421 atkvnfkqvd dlasvwcqcg elelrhenyd ealrllrkat alparraeyf dgsepvqnrv 481 ykslkvwsml adleeslgtf qstkavydri ldlriatpqi vinyamflee hkyfeesfka 541 yergislfkw pnvsdiwsty ltkfiarygg rklerardlf eqaldgcppk yaktlyllya 601 qleeewglar hamavyerat ravepaqqyd mfniyikraa eiygvthtrg iyqkaievls 661 deharemclr fadmecklge idraraiysf csqicdprtt gafwqtwkdf evrhgnedti 721 kemlrirrsv qatyntqvnf masqmlkvsg satgtvsdla pgqsgmddmk lleqraeqla 781 aeaerdqplr aqskilfvrs dasreelael aqqvnpeeiq lgedededem dlepnevrle 841 qqsvpaavfg slked';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count72 = 0; count72 < 4500; count72++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 massasartp agkrvinqee lrrlmkekqr lstsrkries pfakynrlgq lscalcntpv 61 ksellwqthv lgkqhrekva elkgakeasq gssassaphs vkrkapdadd qdvkrakatl 121 vpqvqpstsa wttnfdkigk efiratpskp sglsllpdye deeeeeeeee gdgerkrgda 181 skplsdaqgk ehsvsssrev tssvlpndff stnppkapii phsgsiekae ihekvverre 241 ntaealpegf fddpevdarv rkvdapkdqm dkewdefqka mrqvntisea ivaeedeegr 301 ldrqigeide qiecyrrvek lrnrqdeikn klkeiltike lqkkeeenad sddegelqdl 361 lsqdwrvkga ll';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 magtglvage vvvdalpyfd qgyeapgvre aaaalveeet rryrptknyl syltapdysa 61 fetdimrnef erlaarqpie llsmkryelp apssgqkndi tawqecvnns maqlehqavr 121 ienlelmsqh gcnawkvyne nlvhmiehaq kelqklrkhi qdlnwqrknm qltagsklre 181 mesnwvslvs knyeiertiv qleneiyqik qqhgeanken irqdf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count74 = 0; count74 < 4500; count74++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgrdtrsrsr sagrrgrrrq sqsgsrsrsr shgrrnrrrr edegrrrrrr rsrerrsdse 61 eerwqrsgmr srspprpkwh srdgssqsds geeqsrgqwa rrrrrarsws psssasssas 121 pgrsqspraa aaalsqqqsl qerlrlreer kqqeelmkaf etpeekrarr lakkeakerk 181 krekmgwgee ymgytntdnp fgdnnllgtf iwnkalekkg ishleekelk ernkriqedn 241 rlelqkvkql rlererekam reqelemlqr ekeaehfktw eeqednfhlq qaklrskiri 301 rdgrakpidl lakyisaedd dlavemhepy tflngltvad medllediqv ymeleqgkna 361 dfwrdmttit edeisklrkl easgkgpger regvnasvss dvqsvfkgkt ynqlqvifqg 421 iegkiraggp nldmgywesl lqqlrahmar arlrerhqdv lrqklyklkq eqgveseplf 481 pilkqepqsp srslepedaa ptppgpsseg gpaeaevdga tptegdgdgd gegegegeav 541 lmeedliqqs lddydagrys prlltahelp ldahvlepde dlqrlqlsrq qlqvtgdase 601 saediffrra kegmgqdeaq fsvempltgk aylwadkyrp rkprffnrvh tgfewnkynq 661 thydfdnppp kivqgykfni fypdlidkrs tpeyfleaca dnkdfailrf hagppyedia 721 fkivnrewey shrhgfrcqf angifqlwfh fkryryrr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqdgrkggay agkmeattag vgrleeealr rkerlkalre ktgrkdkedg epktkhlree 61 eeegekhrel rlrnyvpede dlkkrrvpqa kpvaveekvk eqleaakpep vieevdlanl 121 aprkpdwdlk rdvakklekl kkrtqraiae lirerlkgqe dslasavdaa teqktcdsd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count76 = 0; count76 < 4500; count76++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mttaarptfe parggrgkge gdlsqlskqy ssrdlpshtk ikyrqttqda peevrnrdfr 61 releereraa areknrdrpt rehttsssvs kkprldqipa anldaddplt deededfeee 121 sddddtaall aelekikker aeeqarkeqe qkaeeerirm enilsgnpll nltgpsqpqa 181 nfkvkrrwdd dvvfkncakg vddqkkdkrf vndtlrsefh kkfmekyik';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msniyiqepp tngkvllktt agdidielws keapkacrnf iqlcleayyd ntifhrvvpg 61 fivqggdptg tgsggesiyg apfkdefhsr lrfnrrglva managshdng sqffftlgra 121 delnnkhtif gkvtgdtvyn mlrlsevdid dderphnphk ikscevlfnp fddiipreik 181 rlkkekpeee vkklkpkgtk nfsllsfgee aeeeeeevnr vsqsmkgksk sshdllkddp 241 hlssvpvves ekgdapdlvd dgedesaehd eyidgdeknl mreriakklk kdtsanvksa 301 gegevekksv srseelrkea rqlkrellaa kqkkvenaak qaekrseeee appdgavaey 361 rrekqkyeal rkqqskkgts redqtlalln qfkskltqai aetpendipe teveddegwm 421 shvlqfedks rkvkdasmqd sdtfeiydpr npvnkrrree skklmrekke rr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count78 = 0; count78 < 4500; count78++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mpkvvsrsvv csdtrdreey ddgekplhvy yclcgqmvlv ldcqleklpm rprdrsrvid 61 aakhahkfcn tedeetmylr rpegierqyr kkcakcglpl fyqsqpknap vtfivdgavv 121 kfgqgfgktn iytqkqeppk kvmmtkrtkd mgkfssvtvs tideeeeeie arevadsyaq 181 nakviekqle rkgmskrrlq elaeleakka kmkgtlidnq fk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 meeseperkr artdevpagg srseaededd edyvpyvplr qrrqlllqkl lqrrrkgaae 61 eeqqdsgsep rgdeddiplg pqsnvslldq hqhlkekaea rkesakekql keeekilesv 121 aegralmsvk emakgitydd piktswtppr yvlsmseerh ervrkkyhil vegdgipppi 181 ksfkemkfpa ailrglkkkg ihhptpiqiq giptilsgrd migiaftgsg ktlvftlpvi 241 mfcleqekrl pfskregpyg liicpsrela rqthgileyy crllqedssp llrcalcigg 301 msvkeqmeti rhgvhmmvat pgrlmdllqk kmvsldicry laldeadrmi dmgfegdirt 361 ifsyfkgqrq tllfsatmpk kiqnfaksal vkpvtinvgr agaasldviq eveyvkeeak 421 mvylleclqk tpppvlifae kkadvdaihe ylllkgveav aihggkdqee rtkaieafre 481 gkkdvlvatd vaskgldfpa iqhvinydmp eeienyvhri grtgrsgntg iattfinkac 541 desvlmdlka llleakqkvp pvlqvlhcgd esmldigger gcafcgglgh ritdcpklea 601 mqtkqvsnig rkdylahssm df';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count80 = 0; count80 < 4500; count80++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mavavamaga ligsepgpae elakleylsl vskvcteldn hlgindkdla efvislaekn 61 ttfdtfkasl vkngaeftds lisnllrliq tmrppakpst skdpvvkpkt ekeklkelfp 121 vlcqpdnpsv rtmldeddvk vavdvlkele almpsaagqe kqrdaehrdr tkkkkrsrsr 181 drnrdrdrdr ernrdrdhkr rhrsrsrsrs rtrernkvks ryrsrsrsqs ppkdrkdrdk 241 ygernldrwr dkhvdrpppe eptigdiyng kvtsimqfgc fvqleglrkr weglvhisel 301 rregrvanva dvvskgqrvk vkvlsftgtk tslsmkdvdq etgedlnpnr rrnlvgetne 361 etsmrnpdrp thlslvsape veddslerkr ltrisdpekw eikqmiaanv lskeefpdfd 421 eetgilpkvd deededleie lveeeppflr ghtkqsmdms pikivknpdg slsqaammqs 481 alakerrelk qaqreaemds ipmglnkhwv dplpdaegrq iaanmrgigm mpndipewkk 541 hafggnkasy gkktqmsile qreslpiykl keqlvqavhd nqiliviget gsgkttqitq 601 ylaeagytsr gkigctqprr vaamsvakrv seefgcclgq evgytirfed ctspetviky 661 mtdgmllrec lidpdltqya iimldeaher tihtdvlfgl lkktvqkrqd mklivtsatl 721 davkfsqyfy eapiftipgr typveilytk epetdyldas litvmqihlt eppgdilvfl 781 tgqeeidtac eilyermksl gpdvpeliil pvysalpsem qtrifdpapp gsrkvviatn 841 iaetsltidg iyyvvdpgfv kqkvynsktg idqlvvtpis qaqakqragr agrtgpgkcy 901 rlyterayrd emlttnvpei qrtnlastvl slkamgindl lsfdfmdapp metlitameq 961 lytlgaldde glltrlgrrm aefplepmlc kmlimsvhlg cseemltivs mlsvqnvfyr 1021 pkdkqaladq kkakfhqteg dhltllavyn swknnkfsnp wcyenfiqar slrraqdirk 1081 qmlgimdrhk ldvvscgkst vrvqkaicsg ffrnaakkdp qegyrtlidq qvvyihpssa 1141 lfnrqpewvv yhelvlttke ymrevttidp rwlvefapaf fkvsdptkls kqkkqqrlep 1201 lynryeepna wrisrafrrr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maapvgpvkf wrpgtegpgv siseerqsla ensgttvvyn pyaalsieqq rqklpvfklr 61 nhilylieny qtvvivgetg cgkstqipqy laeagwtaeg rvvgvtqprr vaavtvagrv 121 aeergavlgh evgycirfdd ctdqlatrik fltdgmlvre mmvdplltky svimldeahe 181 rtlytdiaig llkkiqkkrg dlrlivasat ldadkfrdff nqnetsdpar dtcviltveg 241 rtfpvdifyl qspvpdyiks tvetvvkihq tegdgdvlaf ltgqeevetv vsmlieqara 301 lartgmkrhl rvlpmyaglp sfeqmkvfer vsrsvrkviv atnvaetsit isgivyvidc 361 gfvklraynp rtaieclvvv pvsqasanqr agrggrsrsg kcyrlyteea fdklpqstvp 421 emqrsnlapv ilqlkalgid nvlrfhfmsp ppaqsmvqal ellyalggld kdcrlteplg 481 mriaefplnp mfakmllesg nfgcsqeils iaammqiqni fvvppnqksh airvhrkfav 541 eegdhltmln iyeafikhnk dskwcqehfl nykglvraat vreqlkkllv kfqvprksse 601 gdpdlvlrci vsgffanaar fhstgayrti rddhelhihp asvlyaekpp rwviyneviq 661 tskyymrdvt aiesawllel aphfyqqgth lslkakrakv qdp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count82 = 0; count82 < 4500; count82++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 meayeqvqkg plklkgvael gvtkrkkkkk dkdkakllea mgtskkneee krrgldkrtp 61 aqaafekmqe krqmerilkk askthkqrve dfnrhldtlt ehydipkvsw tk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maqykgaase agramhlmkk rekqreqmeq mkqriaeeni mksnidkkfs ahydaveael 61 ksstvglvtl ndmkakqeal vkerekqlak keqskelqmk leklrekerk keakrkissl 121 sftleeeeeg geeeeeaamy eeemereeit tkkrklgknp dvdtsflpdr dreeeenrlr 181 eelrqeweak qekikseeie itfsywdgsg hrrtvkmrkg ntmqqflqka leilrkdfse 241 lrsagveqlm yikedliiph hhsfydfivt kargksgplf nfdvhddvrl lsdatvekde 301 shagkvvlrs wyeknkhifp asrwepydpe kkwdkytir';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count84 = 0; count84 < 4500; count84++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhghggydsd fsddercges skrkkrtved dlllqkpfqk ekhgkvahkq vaaelldree 61 arnrrfhlia mdayqrhtkf vndyilyygg kkedfkrlge ndktdldvir enhrflwnee 121 demdmtwekr lakkyydklf keyciadlsk ykenkfgfrw rvekevisgk gqffcgnkyc 181 dkkeglkswe vnfgyiehge krnalvklrl cqecsiklnf hhrrkeiksk krkdktkkdc 241 eesshkksrl ssaeeaskkk dkghssskks edsllrnsde eesaseselw kgplpetdek 301 sqeeefdeyf qdlfl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 masnvtnktd prsmnsrvfi gnlntlvvkk sdveaifsky gkivgcsvhk gfafvqyvne 61 rnaraavage dgrmiagqvl dinlaaepkv nrgkagvkrs aaemygsvte hpspspllss 121 sfdldydfqr dyydrmysyp arvpppppia ravvpskrqr vsgntsrrgk sgfnsksgqr 181 gssksgklkg ddlqaikkel tqikqkvdsl lenlekieke qskqavemkn dkseeeqsss 241 svkkdetnvk meseggadds aeegdllddd dnedrgddql elikddekea eegeddrdsa 301 ngedds';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count86 = 0; count86 < 4500; count86++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MSKGPAVGIDLGTTYSCVGVFQHGKVEIIANDQGNRTTPSYVAF TDTERLIGDAAKNQVAMNPTNTVFDAKRLIGRRFDDAVVQSDMKHWPFMVVNDAGRPK VQVEYKGETKSFYPEEVSSMVLTKMKEIAEAYLGKTVTNAVVTVPAYFNDSQRQATKD AGTIAGLNVLRIINEPTAAAIAYGLDKKVGAERNVLIFDLGGGTFDVSILTIEDGIFE VKSTAGDTHLGGEDFDNRMVNHFIAEFKRKHKKDISENKRAVRRLRTACERAKRTLSS STQASIEIDSLYEGIDFYTSITRARFEELNADLFRGTLDPVEKALRDAKLDKSQIHDI VLVGGSTRIPKIQKLLQDFFNGKELNKSINPDEAVAYGAAVQAAILSGDKSENVQDLL LLDVTPLSLGIETAGGVMTVLIKRNTTIPTKQTQTFTTYSDNQPGVLIQVYEGERAMT KDNNLLGKFELTGIPPAPRGVPQIEVTFDIDANGILNVSAVDKSTGKENKITITNDKG RLSKEDIERMVQEAEKYKAEDEKQRDKVSSKNSLESYAFNMKATVEDEKLQGKINDED KQKILDKCNEIINWLDKNQTAEKEEFEHQQKELEKVCNPIITKLYQSAGGMPGGMPGG FPGGGAPPSGGASSGPTIEEVD';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnilpkkswh vrnkdnvarv rrdeaqaree ekererrvll aqqeartefl rkkarhqnsl 61 peleaaeaga pgsgpvdlfr elleegkgvi rgnkeykeek rqekerqeka lgiltylgqs 121 aaeaqtqppw yqlppgrggp ppgpapdeki ksrldplrem qkhlgkkrqh ggdegsrsrk 181 ekegsekqrp keppsldqlr aerlrreaae rsraeallar vqgralqegq peedetddrr 241 rrynsqfnpq larrprqqdp hlth';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count88 = 0; count88 < 4500; count88++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtrhgkncta gavytyhekk kdtaasgygt qnirlsrdav kdfdccclsl qpchdpvvtp 61 dgylyereai leyilhqkke iarqmkayek qrgtrreeqk elqraasqdh vrgflekesa 121 ivsrplnpft akalsgtspd dvqpgpsvgp pskdkdkvlp sfwipsltpe akatklekps 181 rtvtcpmsgk plrmsdltpv hftpldssvd rvglitrser yvcavtrdsl snatpcavlr 241 psgavvtlec veklirkdmv dpvtgdkltd rdiivlqrgg tgfagsgvkl qaeksrpvmq 301 a';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mveevqkhsv htlvfrslkr thdmfvadng kpvpldeesh krkmaiklrn eygpvlhmpt 61 skenlkekgp qnatdsyvhk qypanqgqev eyfvagthpy ppgpgvalta dtkiqrmpse 121 saaqslaval plqtkadanr tapsgseyrh pgasdrpqpt amnsivmetg ntknsalmak 181 kaptmpkpqw hppwklyrvi sghlgwvrci avepgnqwfv tgsadrtiki wdlasgklkl 241 sltghistvr gvivstrspy lfscgedkqv kcwdleynkv irhyhghlsa vygldlhpti 301 dvlvtcsrds tariwdvrtk asvhtlsght navatvrcqa aepqiitgsh dttirlwdlv 361 agktrvtltn hkksvravvl hprhytfasg spdnikqwkf pdgsfiqnls ghnaiintlt 421 vnsdgvlvsg adngtmhlwd wrtgynfqrv haavqpgsld sesgifacaf dqsesrllta 481 eadktikvyr eddtateeth pvswkpeiik rkrf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count90 = 0; count90 < 4500; count90++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgikvqrprc ffdiainnqp agrvvfelfs dvcpktcenf rclctgekgt gkstqkplhy 61 ksclfhrvvk dfmvqggdfs egngrggesi yggffedesf avkhnkefll smanrgkdtn 121 gsqffittkp tphldghhvv fgqvisgqev vreienqktd aaskpfaevr ilscgelipk 181 skvkkeekkr hkssssssss ssdsdsssds qsssdssdse sateekskkr kkkhrknsrk 241 hkkekkkrkk skksassese aenleaqpqs tvrpeeippi penrflmrks ppkadekerk 301 nrerererec nppnsqpasy qrrllvtrsg rkikgrgprr yrtpsrsrsr drfrrsetpp 361 hwrqemqraq rmrvssgerw ikgdkselne ikenqrspvr vkerkitdhr nvsespnrkn 421 ekekkvkdhk snskerdirr nsekddkykn kvkkraksks rskskeksks kerdskhnrn 481 eekrmrsrsk grdhenvkek ekqsdskgkd qersrskeks kqlesksneh dhskskekdr 541 raqsrsrecd itkgkhsyns rtrersrsrd rsrrvrsrth drdrsrskey hryreqeyrr 601 rgrsrsrerr tppgrsrskd rrrrrrdsrs sereesqsrn kdkyrnqesk sshrkenses 661 ekrmysksrd hnssnnsrek kadrdqspfs kikqssqdne lkssmlknke dekirssvek 721 enqkskgqen dhvheknkkf dhesspgtde dksg';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msvtlhtdvg dikievfcer tpktcemesr cvpqagvqwr dlgslqpppp gfkqvfclsl 61 prtgrggnsi wgkkfedeys eylkhnvrgv vsmanngpnt ngsqffityg kqphldmkyt 121 vfgkvidgle tldeleklpv nektyrplnd vhikditiha npfaq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count92 = 0; count92 < 4500; count92++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maaesgsdfq qrrrrrrdpe epektelser elavavavsq endeeneerw vgplpveatl 61 akkrkvlefe rvyldnlpsa smyersymhr dvithvvctk tdfiitashd ghvkfwkkie 121 egiefvkhfr shlgviesia vssegalfcs vgddkamkvf dvvnfdminm lklgyfpgqc 181 ewiycpgdai ssvaasekst gkifiydgrg dnqplhifdk lhtspltqir lnpvykavvs 241 sdksgmieyw tgppheykfp knvnweyktd tdlyefakck ayptsvcfsp dgkkiatigs 301 drkvrifrfv tgklmrvfde slsmftelqq mrqqlpdmef grrmaverel ekvdavrlin 361 ivfdetghfv lygtmlgikv invetnrcvr ilgkqenirv mqlalfqgia kkhraattie 421 mkasenpvlq niqadptivc tsfkknrfym ftkrepedtk sadsdrdvfn ekpskeevma 481 atqaegpkrv sdsaiihtsm gdihtklfpv ecpktvenfc vhsrngyyng htfhriikgf 541 miqtgdptgt gmggesiwgg efedefhstl rhdrpytlsm anagsntngs qffitvvptp 601 wldnkhtvfg rvtkgmevvq risnvkvnpk tdkpyedvsi initvk';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdilkseilr krqlvedrnl lvenkkyfkr selakkeeea yfercgykiq pkeedqkplt 61 ssnpvlelel aeeklpmtls rqevirrlre rgepirlfge tdydafqrlr kieiltpevn 121 kglrndlkaa ldkidqqyln eivggqepge edtqndlkvh eenttieele algeslgkgd 181 dhkdmdiitk flkfllgvwa kelnaredyv krsvqgklns atqkqtesyl rplfrklrkr 241 nlpadikesi tdiikfmlqr eyvkandayl qmaignapwp igvtmvgiha rtgrekifsk 301 hvahvlndet qrkyiqglkr lmticqkhfp tdpskcveyn al';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count94 = 0; count94 < 4500; count94++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mslicsisne vpehpcvspv snhvyerrli ekyiaengtd pinnqplsee qlidikvahp 61 irpkppsats ipailkalqd ewdavmlhsf tlrqqlqttr qelshalyqh daacrviarl 121 tkevtaarea latlkpqagl ivpqavpssq psvvgagepm dlgelvgmtp eiiqklqdka 181 tvltterkkr gktvpeelvk peelskyrqv ashvglhsas ipgilaldlc psdtnkiltg 241 gadknvvvfd ksseqilatl kghtkkvtsv vfhpsqdlvf saspdatiri wsvpnascvq 301 vvrahesavt glslhatgdy llsssddqyw afsdiqtgrv ltkvtdetsg csltcaqfhp 361 dglifgtgtm dsqikiwdlk ertnvanfpg hsgpitsiaf sengyylata addssvklwd 421 lrklknfktl qldnnfevks lifdqsgtyl alggtdvqiy ickqwteilh ftehsglttg 481 vafghhakfi astgmdrslk fysl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 matslgsnty nrqnwedadf pilcqtclge npyirmtkek ygkeckicar pftvfrwcpg 61 vrmrfkktev cqtcsklknv cqtclldley glpiqvrdag lsfkddmpks dvnkeyytqn 121 mereisnsdg trpvgmlgka tstsdmllkl arttpyykrn rphicsfwvk geckrgeecp 181 yrhekptdpd dpladqnikd ryygindpva dkllkrastm prldppedkt ittlyvgglg 241 dtitetdlrn hfyqfgeirt itvvqrqqca fiqfatrqaa evaaeksfnk livngrrlnv 301 kwgrsqaarg kekekdgttd sgiklepvpg lpgalppppa aeeeasanyf nlppsgppav 361 vnialppppg iapppppgfg phmfhpmgpp ppfmrapgpi hypsqdpqrm gahagkhssp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count96 = 0; count96 < 4500; count96++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeaaalvwi rgpgfgckav rcasgrctvr dfihrhcqdq nvpvenffvk cngalintsd 61 tvqhgavysl eprlcggkgg fgsmlralga qiekttnrea crdlsgrrlr dvnhekamae 121 wvkqqaerea ekeqkrlerl qrklvepkhc ftspdyqqqc hemaerleds vlkgmqaass 181 kmvsaeisen rkrqwptksq tdrgasagkr rcfwlgmegl etaegsnses sdddseeaps 241 tsgmgfhapk igsngvemaa kfpsgsqrar vvntdhgspe qlqipvtdsg rhiledscae 301 lgeskehmes rmvteteetq ekkaeskepi eeeptgagln kdketeertd gervaevape 361 erenvavakl qesqpgnavi dketidllaf tsvaelellg leklkcelma lglkcggtlq 421 eraarlfsvr glakeqidpa lfakplkgkk k';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdlglaaagl ggqitmsatv vdavnaapls gskemsleep kkmtredwrk kkeleeqrkl 61 gnapaevdee gkdinphipq yissvpwyid pskrptlkhq rpqpekqkqf sssgewykrg 121 vkensiitky rkgacencga mthkkkdcfe rprrvgakft gtniapdehv qpqlmfdydg 181 krdrwngynp eehmkiveey akvdlakrtl kaqklqeela sgklveqans pkhqwgeeep 241 nsqmekdhns edededkyad didmpgqnfd skrritvrnl rirediakyl rnldpnsayy 301 dpktramren pyanagknpd evsyagdnfv rytgdtisma qtqlfaweay dkgsevhlqa 361 dptklellyk sfkvkkedfk eqqkesilek yggqehldap paelllaqte dyveysrhgt 421 vikgqerava cskyeedvki hnhthiwgsy wkegrwgykc chsffkysyc tgeagkeivn 481 seeciineit geesvkkpqt lmelhqeklk eekkkkkkkk kkhrksssds ddeekkhekl 541 kkalnaeear llhvketmqi derkrpynsm yetrepteee meayrmkrqr pddpmasflg 601 q';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

for (var count98 = 0; count98 < 4500; count98++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdagffrgts aeqdnrfsnk qkkllkqlkf aeclekkvdm skvnlevikp witkrvteil 61 gfeddvvief ifnqlevknp dskmmqinlt gflngknare fmgelwplll saqeniagip 121 saflelkkee ikqrqieqek lasmkkqded kdkrdkeeke ssrekrersr sprrrksrsp 181 sprrrsspvr rerkrshsrs prhrtksrsp spapekkekt pelpepsvkv kepsvqeats 241 tsdilkvpkp epipepkeps peknskkeke kektrprsrs rsksrsrtrs rspshtrprr 301 rhrsrsrsys prrrpsprrr psprrrtppr rmpppprhrr srspvrrrrr ssaslsgsss 361 sssssrsrsp pkkppkrtss pprktrrlsp saspprrrhr psppatpppk trhsptpqqs 421 nrtrksrvsv spgrtsvtkh kgtekresps papkprkvel seseedkggk maaadsvqqr 481 rqyrrqnqqs ssdsgsssss ederpkrshv kngevgrrrr hspsrsasps prkrqketsp 541 rmqmgkrwqs pvtksgrrrr spsppptrrr rspspapppr rrrtptpppr rrtpsppprr 601 rspsprrysp piqrryspsp ppkrrtaspp pppkrrasps pppkrrvshs pppkqrsspv 661 tkrrspslss khrkgsspsr strearspqp nkrhspsprp rapqtssspp pvrrgasssp 721 qrrqspspst rpirrvsrtp epkkikkaas pspqsvrrvs ssrsvsgspe paakkppapp 781 spvqsqspst nwspavpvkk aksptpspsp prnsdqeggg kkkkkkkdkk hkkdkkhkkh 841 kkhkkekava aaaaaavtpa aiaaatttla qeepvaapep kketeseaed nlddlekhlr 901 ekalrsmrka qvspqs';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdalesllde valegldglc lpalwsrlet rvppfplple pctqeflwra lathpgisfy 61 eeprerpdlq lqdryeeidl etgilesrrd pvaledvypi hmilenkdgi qgscryfker 121 knitndirtk slqprctmve afdrwgkkli ivasqamryr aligqegdpd lklpdfsyci 181 lerlgrsrwq gelqrdlhtt afkvdagklh yhrkilnkng litmqshvir lptgaqqhsi 241 llllnrfhvd rrskydilme klsvmlstrt nhietlgklr eelglcertf krlyqymlna 301 glakvvslrl qeihpecgpc ktkkgtdvmv rclkllkefk rndhdddede evisktvppv 361 divferdmlt qtydlierrg tkgisqaeir vamnvgklea rmlcrllqrf kvvkgfmede 421 grqrttkyis cvfaeesdls rqyqrekars ellttvslas mqeesllpeg edtflsesds 481 eeerssskrr grgsqkdtra sanlrpktqp hhstptkggw kvvnlhplkk qppsfpgaae 541 eracqslasr dslldtssvs epnvsfvshc adsnsgdiav ieevrmenpk esssslktgr 601 hssgqdkphe tyrllkrrnl iieavtnlrl ieslftiqkm imdqekqegv stkcckksiv 661 rlvrnlseeg llrlyrttvi qdgikkkvdl vvhpsmdqnd plvrsaieqv rfrisnssta 721 nrvktsqppv pqgeaeedsq gkegpsgsgd sqlsassrse sgrmkksdnk mgitplrnyh 781 pivvpglgrs lgflpkmprl rvvhmflwyl iyghpasntv ekpsfiserr tikqesgrag 841 vrpsssgsaw eacseapskg sqdgvtweae velatetvyv ddaswmryip pipvhrdfgf 901 gwalvsdill clplsifiqi vqvsykvdnl eeflndplkk htlirflprp irqqllykrr 961 yifsvvenlq rlcymgllqf gptekfqdkd qvfiflkkna vivdtticdp hynlarssrp 1021 ferrlyvlns mqdvenywfd lqcvclntpl gvvrcprvrk nsstdqgsde egslqkeqes 1081 amdkhnlerk camleyttgs revvdeglip gdglgaagld ssfyghlkrn wiwtsyiinq 1141 akkentaaen gltvrlqtfl skrpmplsar gnsrlniwge arvgselcag weeqfevdre 1201 psldrnrrvr ggksqkrkrl kkdpgkkikr kkkgefpgek skrlryhdea dqsalqrmtr 1261 lrvtwsmqed gllvlcrias nvlntkvkgp fvtwqvvrdi lhatfeesld ktshsvgrra 1321 ryivknpqay lnykvclaev yqdkalvgdf mnrrgdyddp kvcanefkef veklkekfss 1381 alrnsnleip dtlqelfary rvlaigdekd qtrkedelns vddihflvlq nliqstlals 1441 dsqmksyqsf qtfrlyreyk dhvlvkafme cqkrslvnrr rvnhtlgpkk nralpfvpms 1501 yqlsqtyyri ftwrfpstic tesfqfldrm raagkldqpd rfsfkdqdnn eptndmvafs 1561 ldgpggncva vltlfslgli svdvripeqi ivvdssmven evikslgkdg sleddedeed 1621 dldegvggkr rsmevkpaqa shtnyllmrg yyspgivstr nlnpndsivv nscqmkfqlr 1681 ctpvparlrp aaapleeltm gtsclpdtft klinpqentc sleefvlqle lsgyspedlt 1741 aaleileaii atgcfgidke elrrrfsale kagggrtrtf adciqalleq hqvlevggnt 1801 arlvamgsaw pwllhsvrlk dredadiqre dpqarplegs ssedsppegq appshsprgt 1861 krraswasen getdaegtqm tpakrpalqd snlapslgpg aedgaeaqap spppaledta 1921 aagaaqedqe gvgefsspgq eqlsgqaqpp egsedprgft esfgaanisq aarerdcesv 1981 cfigrpwrvv dghlnlpvck gmmeamlyhi mtrpgipess llrhyqgvlq pvavlellqg 2041 leslgcirkr wlrkprpvsl fstpvveeve vpssldespm afyeptldct lrlgrvfphe 2101 vnwnkwihl';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count100 = 0; count100 < 4500; count100++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MDTCGVGYVALGEAGPVGNMTVVDSPGQEVLNQLDVKTSSEMTS AEASVEMSLPTPLPGFEDSPDQRRLPPEQESLSRLEQPDLSSEMSKVSKPRASKPGRK RGGRTRKGPKRPQQPNPPSAPLVPGLLDQSNPLSTPMPKKRGRKSKAELLLLKLSKDL DRPESQSPKRPPEDFETPSGERPRRRAAQVALLYLQELAEELSTALPAPVSCPEGPKV SSPTKPKKIRQPAACPGGEEVDGAPRDEDFFLQVEAEDVEESEGPSESSSEPEPVVPR STPRGSTSGKQKPHCRGMAPNGLPNHIMAPVWKCLHLTKDFREQKHSYWEFAEWIPLA WKWHLLSELEAAPYLPQEEKSPLFSVQREGLPEDGTLYRINRFSSITAHPERWDVSFF TGGPLWALDWCPVPEGAGASQYVALFSSPDMNETHPLSQLHSGPGLLQLWGLGTLQQE SCPGNRAHFVYGIACDNGCIWDLKFCPSGAWELPGTPRKAPLLPRLGLLALACSDGKV LLFSLPHPEALLAQQPPDAVKPAIYKVQCVATLQVGSMQATDPSECGQCLSLAWMPTR PHQHLAAGYYNGMVVFWNLPTNSPLQRIRLSDGSLKLYPFQCFLAHDQAVRTLQWCKA NSHFLVSAGSDRKIKFWDLRRPYEPINSIKRFLSTELAWLLPYNGVTVAQDNCYASYG LCGIHYIDAGYLGFKAYFTAPRKGTVWSLSGSDWLGTIAAGDISGELIAAILPDMALN PINVKRPVERRFPIYKADLIPYQDSPEGPDHSSASSGVPNPPKARTYTETVNHHYLLF QDTDLGSFHDLLRREPMLRMQEGEGHSQLCLDRLQLEAIHKVRFSPNLDSYGWLVSGG QSGLVRIHFVRGLASPLGHRMQLESRAHFNAMFQPSSPTRRPGFSPTSHRLLPTP';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msgfspelid ylegkisfee ferrreerkt rekkslqekg klsaeenpdd sevpsssgin 61 stksqdkdvn egetsdgvrk svhkvfasml geneddeeee eeeeeeeeee etpeqptagd 121 vfvlemvlnr etkkmmkekr prsklpralr glmgeanirf argereeail mcmeiirqap 181 layepfstla miyedqgdme kslqfeliaa hlnpsdteew vrlaemsleq dnikqaifcy 241 tkalkyeptn vrylwerssl yeqmgdhkma mdgyrrilnl lspsdgerfm qlardmaksy 301 yeandvtsai niideafskh qglvsmedvn iaaelyisnk qydkaleiit dfsgivlekk 361 tseegtseen kapenvtcti pdgvpiditv klmvclvhln ileplnpllt tlveqnpedm 421 gdlyldvaea fldvgeynsa lpllsalvcs erynlavvwl rhaeclkalg ymeraaesyg 481 kvvdlaplhl darislstlq qqlgqpekal ealepmydpd tlaqdanaaq qelklllhrs 541 tllfsqgkmy gyvdtlltml amllkvamnr aqvclisssk sgerhlylik vsrdkisdsn 601 dqesancdak aifavltsvl tkddwwnlll kaiyslcdls rfqeaellvd ssleyysfyd 661 drqkrkeley fglsaaildk nfrkaynyir imvmenvnkp qlwnifnqvt mhsqdvrhhr 721 fclrlmlknp enhalcvlng hnafvsgsfk halgqyvqaf rthpdeplys fcigltfihm 781 asqkyvlrrh alivqgfsfl nrylslrgpc qesfynlgrg lhqlglihla ihyyqkalel 841 pplvvegiel dqldlrrdia ynlsliyqss gntgmaqtll ytycsi';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count102 = 0; count102 < 4500; count102++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mntadqarvg paddgpapsg eeegegggea ggkepaadaa pgpsaafrlm vtrrepavkl 61 qyavsglepl awsedhrvsv starsiavle licdvhnpgq dlvihrtsvp aplnscllkv 121 gsktevaeck ekfaaskdpt vsqtfmldrv fnpegkalpp mrgfkytsws pmgcdangrc 181 llaaltmdnr ltiqanlnrl qwvqlvdlte iygerlyets yrlskneape gnlgdfaefq 241 rrhsmqtpvr mewsgicttq qvkhnnecrd vgsvllavlf engniavwqf qlpfvgkesi 301 sscntiesgi tspsvlfwwe yehnnrkmsg livgsafgpi kilpvnlkav kgyftlrqpv 361 ilwkemdqlp vhsikcvply hpyqkcscsl vvaargsyvf wcllliskag lnvhnshvtg 421 lhslpivsmt adkqngtvyt cssdgkvrql ipiftdvalk fehqliklsd vfgsvrthgi 481 avspcgayla iittegming lhpvnknyqv qfvtlktfee aaaqllessv qnlfkqvdli 541 dlvrwkilkd khipqflqea lekkiessgv tyfwrfklfl lrilyqsmqk tpsealwkpt 601 hedskillvd spgmgnadde qqeegtsskq vvkqglqers kegdveeptd dslpttgdag 661 grepmeekll eiqgkieave mhltrehmkr vlgevylhtw itentsiptr glcnflmsde 721 eyddrtarvl ighiskkmnk qtfpehcslc keilpftdrk qavcsnghiw lrcfltyqsc 781 qsliyrrcll hdsiarhpap edpdwikrll qspcpfcdsp vf';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maaeaadlgl gaavpvelrr errmvcveyp gvvrdvakml ptlggeegvs riyadptkrl 61 elyfrpkdpy chpvcanrfs tsslllrirk rtrrqkgvlg teahsevtfd meilgiisti 121 ykfqgmsdfq ylavhteagg khtsmydkvl mlrpekeaff hqelplyipp pifsrldapv 181 dyfyrpetqh regynnppis genliglsra rrphnaifvn fedeevpkqp leaaaqtwrr 241 vctnpvdrkv eeelrklfdi rpiwsrnavk anisvhpdkl kvllpfiayy mitgpwrslw 301 irfgydprkn pdakiyqvld frircgmkhg yapsdlpvka krstynyslp itvkktssql 361 vtmhdlkqgl gpsgtsgark passkyklkv slqtlrdsvy ifregalppy rqmfyqlcdl 421 nveelqkiih rndgaensct erdgwclpkt sdelrdtmsl mirqtirskr palfsssaka 481 dggkeqltye sgedeedeee eeeeeedfkp sdgsenemet eildyv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count104 = 0; count104 < 4500; count104++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaaadersp edgedeeeee qlvlvelsgi idsdflskce nkckvlgidt erpilqvdsc 61 vfageyedtl gtcvifeenv ehadtegnnk tvlkykchtm kklsmtrtll tekkegeeni 121 ggvewlqikd ndfsyrpnmi cnflhenede evvasapdks leleeeeiqm ndssnlsceq 181 ekpmhleied sgplidipse tegsvfmetq mlp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtgrvcrgcg gtdieldaar gdavctacgs vledniivse vqfvessggg ssavgqfvsl 61 dgagktptlg ggfhvnlgke sraqtlqngr rhihhlgnql qlnqhcldta fnffkmavsr 121 hltrgrkmah viaaclylvc rtegtphmll dlsdllqvnv yvlgktflll arelcinapa 181 idpclyiprf ahllefgekn hevsmtalrl lqrmkrdwmh tgrrpsglcg aallvaarmh 241 dfrrtvkevi svvkvcestl rkrltefedt ptsqltidef mkidleeecd ppsytagqrk 301 lrmkqleqvl skkleevege issyqdaiei elensrpkak gglaslakdg stedtasslc 361 geedtedeel eaaashlnkd lyrellggap gsseaagspe wggrppalgs lldplptaas 421 lgisdsirec issqssdpkd asgdgeldls giddleidry ilnesearvk aelwmrenae 481 ylreqrekea riakekelgi ykehkpkksc krrepiqast areaiekmle qkkisskiny 541 svlrglssag ggsphredaq pehsasarkl srrrtpasrs gadpvtsvgk rlrplvstqp 601 akkvatgeal lpssptlgae parpqavlve sgpvsyhade eadeeepdee dgepcvsalq 661 mmgsndygcd gdeddgy';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count106 = 0; count106 < 4500; count106++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mpgrgrcpdc gstelvedsh ysqsqlvcsd cgcvvtegvl tttfsdegnl revtysrstg 61 eneqvsrsqq rglrrvrdlc rvlqlpptfe dtavayyqqa yrhsgiraar lqkkevlvgc 121 cvlitcrqhn wpltmgaict llyadldvfs stymqivkll gldvpslcla elvktycssf 181 klfqaspsvp akyvedkekm lsrtmqlvel anetwlvtgr hplpvitaat flawqslqpa 241 drlscslarf cklanvdlpy passrlqell avllrmaeql awlrvlrldk rsvvkhigdl 301 lqhrqslvrs afrdgtaeve trekeppgwg qgqgegevgn nslglpqgkr paspalllpp 361 cmlkspkric pvppvstvtg denisdseie qylrtpqevr dfqraqaarq aatsvpnpp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mevkdansal lsnyevfqll tdlkeqrkes gknkhssgqq nlntityetl kyisktpcrh 61 qspeivrefl talkshkltk aeklqllnhr pvtaveiqlm veeseerlte eqieallhtv 121 tsilpaepea eqkkntnsnv amdeedpa';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count108 = 0; count108 < 4500; count108++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mllfcpgcgn gliveegqrc hrfacntcpy vhnitrkvtn rkypklkevd dvlggaaawe 61 nvdstaescp kcehprayfm qlqtrsadep mttfykccna qcghrwrd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count109 = 0; count109 < 4500; count109++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfvlvemvdt vrippwqfer klndsiaeel nkklankvvy nvglciclfd itkledayvf 61 pgdgashtkv hfrcvvfhpf ldeiligkik gcspegvhvs lgffddilip peslqqpakf 121 deaeqvwvwe yeteegahdl ymdtgeeirf rvvdesfvdt sptgpssada ttsseelpkk 181 eapytlvgsi sepglgllsw wtsn';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count110 = 0; count110 < 4500; count110++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 masrgggrgr grgqltfnve avgigkgdal ppptlqpspl fpplefrpvp lpsgeegeyv 61 lalkqelrga mrqlpyfirp avpkrdvery sdkyqmsgpi dnaidwnpdw rrlprelkir 121 vrklqkerit illpkrppkt tedkeetiqk letlekkeee vtseedeeke eeeekeeeee 181 eeydeeehee etdyimsyfd ngedfggdsd dnmdeaiy';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count111 = 0; count111 < 4500; count111++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 magnkgrgra aytfnieavg fskgeklpdv vlkppplfpd tdykpvplkt gegeeymlal 61 kqelretmkr mpyfietpee rqdieryskr ymkvykeewi pdwrrlprem mprnkckkag 121 pkpkkakdag kgtpltnted vlkkmeelek rgdgeksdee neekegskek skegdddddd 181 daaeqeeyde eeqeeendyi nsyfedgddf gadsddnmde aty';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count112 = 0; count112 < 4500; count112++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maevkvkvqp pdadpveien riielchqfp hgitdqviqn emphieaqqr avainrllsm 61 gqldllrsnt gllyrikdsq nagkmkgsdn qeklvyqiie dagnkgiwsr diryksnlpl 121 teinkilknl eskklikavk svaaskkkvy mlynlqpdrs vtggawysdq dfesefvevl 181 nqqcfkflqs kaetareskq npmiqrnssf asshevwkyi celgiskvel smedietiln 241 tliydgkvem tiiaakegtv gsvdghmkly ravnpiippt glvrapcglc pvfddchegg 301 eispsnciym tewlef';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count113 = 0; count113 < 4500; count113++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maneeddpvv qeidvylaks laeklylfqy pvrpasmtyd diphlsakik pkqqkvelem 61 aidtlnpnyc rskgeqialn vdgacadets tyssklmdkq tfcssqttsn tsryaaalyr 121 qgelhltplh gilqlrpsfs yldkadakhr ereaaneagd ssqdeaeddv kqitvrfsrp 181 eseqarqrrv qsyeflqkkh aeepwvhlhy yglrdsrseh erqyllcpgs sgventelvk 241 spseylmmlm ppsqeeekdk pvapsnvlsm aqlrtlplad qikilmknvk vmpfanlmsl 301 lgpsidsvav lrgiqkvaml vqgnwvvksd ilypkdsssp hsgvpaevlc rgrdfvmwkf 361 tqsrwvvrke vatvtklcae dvkdflehma vvrinkgwef ilpydgefik khpdvvqrqh 421 mlwtgiqakl ekvynlvket mpkkpdaqsg paglvcgdqr iqvaktkaqq nhallerelq 481 rrkeqlrvpa vppgvrikee pvseegeede eqeaeeepmd tspsglhskl anglplgraa 541 gtdsfnghpp qgcastpvar elkafveatf qrqfvltlse lkrlfnlhla slppghtlfs 601 gisdrmlqdt vlaagckqil vpfppqtaas pdeqkvfalw esgdmsdqhr qvlleifskn 661 yrvrrnmiqs rltqecgedl skqevdkvlk dccvsyggmw ylkgtvqs';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count114 = 0; count114 < 4500; count114++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msegnaagep stpggprpll tgargligrr pappltpgrl psirsrdltl ggvkkktftp 61 niisrkikee pkeevtvkke krerdrdrqr eghgrgrgrp eviqshsife qgpaemmkkk 121 gnwdktvdvs dmgpshiini kkekretdee tkqilrmlek ddflddpglr ndtrnmpvql 181 plahsgwlfk eendepdvkp wlagpkeedm evdipavkvk eeprdeeeea kmkappkaar 241 ktpglpkdvs vaellrelsl tkeeellflq lpdtlpgqpp tqdikpikte vqgedgqvvl 301 ikqekdreak laenactlad ltegqvgkll irksgrvqll lgkvtldvtm gtacsflqel 361 vsvglgdsrt gemtvlghvk hklvcspdfe slldhkhr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count115 = 0; count115 < 4500; count115++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msegnaagep stpggprpll tgargligrr pappltpgrl psirsrdltl ggvkkktftp 61 niisrkikee pkeevtvkke krerdrdrqr eghgrgrgrp eviqshsife qgpaemmkkk 121 gnwdktvdvs dmgpshiini kkekretdee tkqilrmlek ddflddpglr ndtrnmpvql 181 plahsgwlfk eendepdvkp wlagpkeedm evdipavkvk eeprdeeeea kmkappkaar 241 ktpglpkdvs vaellrelsl tkeeellflq lpdtlpgqpp tqdikpikte vqgedgqvvl 301 ikqekdreak laenactlad ltegqvgkll irksgrvqll lgkvtldvtm gtacsflqel 361 vsvglgdsrt gemtvlghvk hklvcspdfe slldhkhr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count116 = 0; count116 < 4500; count116++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtqaeiklcs lllqehfgei vekigvhlir tgsqplrvia hdtgtsldqv kkalcvlvqh 61 nlvsyqvhkr gvveyeaqcs rvlrmlrypr yiyttktlys dtgeliveel llngkltmsa 121 vvkkvadrlt etmedgktmd yaevsntfvr ladthfvqrc psvpttensd pgppppaptl 181 vinekdmylv pklsligkgk rrrssdedaa gepkakrpky ttdnkepipd dgiywqanld 241 rfhqhfrdqa ivsavanrmd qtsseivrtm lrmseittss sapftqplss neifrslpvg 301 yniskqvldq yltlladdpl efvgksgdsg ggmyvinlhk alaslatatl esvvqerfgs 361 rcarifrlvl qkkhieqkqv edfamipake akdmlykmls enfmslqeip ktpdhapsrt 421 fylytvnils aarmllhrcy ksianlierr qfetkenkrl leksqrveai iasmqatgae 481 eaqlqeieem itaperqqle tlkrnvnkld aseiqvdeti fllesyiect mkrq';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count117 = 0; count117 < 4500; count117++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdvlaeefgn ltpeqlaapi ptveekwrll paflkvkglv kqhidsfnyf inveikkimk 61 anekvtsdad pmwylkylni yvglpdvees fnvtrpvsph ecrlrdmtys apitvdieyt 121 rgsqriirna lpigrmpiml rssncvltgk tpaefaklne cpldpggyfi vkgvekvili 181 qeqlsknrii veadrkgavg asvtssthek ksrtnmavkq grfylrhntl sedipiviif 241 kamgvesdqe ivqmigteeh vmaafgpsle ecqkaqiftq mqalkyignk vrrqrmwggg 301 pkktkieear ellastilth vpvkefnfra kciytavmvr rvilaqgdnk vddrdyygnk 361 rlelagqlls llfedlfkkf nsemkkiadq vipkqraaqf dvvkhmrqdq itngmvnais 421 tgnwslkrfk mdrqgvtqvl srlsyisalg mmtrissqfe ktrkvsgprs lqpsqwgmlc 481 psdtpegeac glvknlalmt hittdmedgp ivklasnlgv edvnllcgee lsypnvflvf 541 lngnilgvir dhkklvntfr lmrragyine fvsistnltd rcvyissdgg rlcrpyiivk 601 kqkpavtnkh meelaqgyrn fedflheslv eyldvneend cnialyehti nkdtthleie 661 pftllgvcag lipyphhnqs prntyqcamg kqamgtigyn qrnridtlmy llaypqkpmv 721 ktktielief eklpagqnat vavmsysgyd iedalvlnka sldrgfgrcl vyknakctlk 781 rytnqtfdkv mgpmldaatr kpiwrheild adgicspgek venkqvlvnk smptvtqipl 841 egsnvpqqpq ykdvpitykg atdsyiekvm issnaedafl ikmllrqtrr peigdkfssr 901 hgqkgvcgli vpqedmpfcd sgicpdiimn phgfpsrmtv gkliellagk agvldgrfhy 961 gtafggskvk dvcedlvrhg ynylgkdyvt sgitgeplea yiyfgpvyyq klkhmvldkm 1021 harargprav ltrqptegrs rdgglrlgem erdcligyga smlllerlmi ssdafevdvc 1081 gqcgllgysg wchyckssch vsslripyac kllfqelqsm niiprlklsk yne';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count118 = 0; count118 < 4500; count118++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvkeqfretd vakkishicf gmkspeemrq qahiqvvskn lysqdnqhap llygvldhrm 61 gtsekdrpce tcgknladcl ghygyidlel pcfhvgyfra vigilqmick tcchimlsqe 121 ekkqfldylk rpgltylqkr glkkkisdkc rkknichhcg afngtvkkcg llkiihekyk 181 tnkkvvdpiv snflqsfeta iehnkevepl lgraqenlnp lvvlnlfkri paedvplllm 241 npeagkpsdl iltrllvppl cirpsvvsdl ksgtneddlt mklteiifln dvikkhrisg 301 aktqmimedw dflqlqcaly inselsgipl nmapkkwtrg fvqrlkgkqg rfrgnlsgkr 361 vdfsgrtvis pdpnlridev avpvhvakil tfpekvnkan inflrklvqn gpevhpganf 421 iqqrhtqmkr flkygnrekm aqelkygdiv erhlidgdvv lfnrqpslhk lsimahlarv 481 kphrtfrfne cvctpynadf dgdemnlhlp qteeakaeal vlmgtkanlv tprngeplia 541 aiqdfltgay lltlkdtffd rakacqiias ilvgkdekik vrlppptilk pvtlwtgkqi 601 fsvilrpsdd npvranlrtk gkqycgkged lcandsyvti qnselmsgsm dkgtlgsgsk 661 nnifyillrd wgqneaadam srlarlapvy lsnrgfsigi gdvtpgqgll kakyellnag 721 ykkcdeyiea lntgklqqqp gctaeetlea lilkelsvir dhagsaclre ldksnspltm 781 alcgskgsfi nisqmiacvg qqaisgsrvp dgfenrslph fekhsklpaa kgfvansfys 841 gltptefffh tmagreglvd tavktaetgy mqrrlvksle dlcsqydltv rsstgdiiqf 901 iyggdgldpa amegkdeple fkrvldnika vfpcpsepal skneliltte simkkseflc 961 cqdsflqeik kfikgvseki kktrdkygin dngtteprvl yqldritptq vekfletcrd 1021 kymraqmepg savgalcaqs igepgtqmtl ktfhfagvas mnitlgvpri keiinaskai 1081 stpiitaqld kdddadyarl vkgriektll geiseyieev flpddcfilv klslerirll 1141 rlevnaetvr ysictsklrv kpgdvavhge avvcvtpren skssmyyvlq flkedlpkvv 1201 vqgipevsra vihideqsgk ekykllvegd nlravmathg vkgtrttsnn tyevektlgi 1261 eaarttiine iqytmvnhgm sidrrhvmll sdlmtykgev lgitrfglak mkesvlmlas 1321 fektadhlfd aayfgqkdsv cgvseciimg ipmnigtglf kllhkadrdp nppkrplifd 1381 tnefhiplvt';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count119 = 0; count119 < 4500; count119++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 miipvrcftc gkivgnkwea ylgllqaeyt egdaldalgl kryccrrmll ahvdliekll 61 nyaplek';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count120 = 0; count120 < 4500; count120++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdtqkdvqpp kqqpmiyicg echteneiks rdpircrecg yrimykkrtk rlvvfdar';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count121 = 0; count121 < 4500; count121++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 magilfedif dvkdidpegk kfdrvsrlhc esesfkmdli ldvniqiypv dlgdkfrlvi 61 astlyedgtl ddgeynptdd rpsradqfey vmygkvyrie gdetsteaat rllrlraaew 121 qcsritgwgl lfqlcvrvlw gpaheaaggc qqpawirggl qslspdeeas llnla';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count122 = 0; count122 < 4500; count122++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 maasqaveem rsrvvlgefg vrnvhttdfp gnysgyddaw dqdrfeknfr vdvvhmdens 61 lefdmvgida aianafrril laevptmave kvlvynntsi vqdeilahrl glipihadpr 121 lfeyrnqgde egteidtlqf rlqvrctrnp haakdssdpn elyvnhkvyt rhmtwiplgn 181 qadlfpegti rpvhddilia qlrpgqeidl lmhcvkgigk dhakfspvat asyrllpdit 241 llepvegeaa eelsrcfspg vievqevqgk kvarvanprl dtfsreifrn eklkkvvrla 301 rvrdhyifsv estgvlppdv lvseaikvlm gkcrrfldel davqmd';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

vital\_components\_2b();

}

function vital\_components\_2b() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = 'MYNGIGLPTPRGSGTNGYVQRNLSLVRGRRGERPDYKGEEELRR LEAALVKRPNPDILDHERKRRVELRCLELEEMMEEQGYEEQQIQEKVATFRLMLLEKD VNPGGKEETPGQRPAVTETHQLAELNEKKNERLRAAFGISDSYVDGSSFDPQRRAREA KQPAPEPPKPYSLVRESSSSRSPTPKQKKKKKKKDRGRRSESSSPRRERKKSSKKKKH RSESESKKRKHRSPTPKSKRKSKDKKRKRSRSTTPAPKSRRAHRSTSADSASSSDTSR SRSRSAAAKTHTTALAGRSPSPASGRRGEGDAPFSEPGTTSTQRPSSPETATKQPSSP YEDKDKDKKEKSATRPSPSPERSSTGPEPPAPTPLLAERHGGSPQPLATTPLSQEPVN PPSEASPTRDRSPPKSPEKLPQSSSSESSPPSPQPTKVSRHASSSPESPKPAPAPGSH REISSSPTSKNRSHGRAKRDKSHSHTPSRRMGRSRSPATAKRGRSRSRTPTKRGHSRS RSPQWRRSRSAQRWGRSRSPQRRGRSRSPQRPGWSRSRNTQRRGRSRSARRGRSHSRS PATRGRSRSRTPARRGRSRSRTPARRRSRSRTPTRRRSRSRTPARRGRSRSRTPARRR SRTRSPVRRRSRSRSPARRSGRSRSRTPARRGRSRSRTPARRGRSRSRTPARRSGRSR SRTPARRGRSRSRTPRRGRSRSRSLVRRGRSHSRTPQRRGRSGSSSERKNKSRTSQRR SRSNSSPEMKKSRISSRRSRSLSSPRSKAKSRLSLRRSLSGSSPCPKQKSQTPPRRSR SGSSQPKAKSRTPPRRSRSSSSPPPKQKSKTPSRQSHSSSSPHPKVKSGTPPRQGSIT SPQANEQSVTPQRRSCFESSPDPELKSRTPSRHSCSGSSPPRVKSSTPPRQSPSRSSS PQPKVKAIISPRQRSHSGSSSPSPSRVTSRTTPRRSRSVSPCSNVESRLLPRYSHSGS SSPDTKVKPETPPRQSHSGSISPYPKVKAQTPPGPSLSGSKSPCPQEKSKDSLVQSCP GSLSLCAGVKSSTPPGESYFGVSSLQLKGQSQTSPDHRSDTSSPEVRQSHSESPSLQS KSQTSPKGGRSRSSSPVTELASRSPIRQDRGEFSASPMLKSGMSPEQSRFQSDSSSYP TVDSNSLLGQSRLETAESKEKMALPPQEDATASPPRQKDKFSPFPVQDRPESSLVFKD TLRTPPRERSGAGSSPETKEQNSALPTSSQDEELMEVVEKSEEPAGQILSHLSSELKE MSTSNFESSPEVEERPAVSLTLDQSQSQASLEAVEVPSMASSWGGPHFSPEHKELSNS PLRENSFGSPLEFRNSGPLGTEMNTGFSSEVKEDLNGPFLNQLETDPSLDMKEQSTRS SGHSSSELSPDAVEKAGMSSNQSISSPVLDAVPRTPSRERSSSASSPEMKDGLPRTPS RRSRSGSSPGLRDGSGTPSRHSLSGSSPGMKDIPRTPSRGRSECDSSPEPKALPQTPR PRSRSPSSPELNNKCLTPQRERSGSESSVDQKTVARTPLGQRSRSGSSQELDVKPSAS PQERSESDSSPDSKAKTRTPLRQRSRSGSSPEVDSKSRLSPRRSRSGSSPEVKDKPRA APRAQSGSDSSPEPKAPAPRALPRRSRSGSSSKGRGPSPEGSSSTESSPEHPPKSRTA RRGSRSSPEPKTKSRTPPRRRSSRSSPELTRKARLSRRSRSASSSPETRSRTPPRHRR SPSVSSPEPAEKSRSSRRRRSASSPRTKTTSRRGRSPSPKPRGLQRSRSRSRREKTRT TRRRDRSGSSQSTSRRRQRSRSRSRVTRRRRGGSGYHSRSPARQESSRTSSRRRRGRS RTPPTSRKRSRSRTSPAPWKRSRSRASPATHRRSRSRTPLISRRRSRSRTSPVSRRRS RSRTSVTRRRSRSRASPVSRRRSRSRTPPVTRRRSRSRTPTTRRRSRSRTPPVTRRRS RSRTPPVTRRRSRSRTSPITRRRSRSRTSPVTRRRSRSRTSPVTRRRSRSRTSPVTRR RSRSRTPPAIRRRSRSRTPLLPRKRSRSRSPLAIRRRSRSRTPRTARGKRSLTRSPPA IRRRSASGSSSDRSRSATPPATRNHSGSRTPPVALNSSRMSCFSRPSMSPTPLDRCRS PGMLEPLGSSRTPMSVLQQAGGSMMDGPGPRIPDHQRTSVPENHAQSRIALALTAISL GTARPPPSMSAAGLAARMSQVPAPVPLMSLRTAPAANLASRIPAASAAAMNLASARTP AIPTAVNLADSRTPAAAAAMNLASPRTAVAPSAVNLADPRTPTAPAVNLAGARTPAAL AALSLTGSGTPPTAANYPSSSRTPQAPASANLVGPRSAHATAPVNIAGSRTAAALAPA SLTSARMAPALSGANLTSPRVPLSAYERVSGRTSPPLLDRARSRTPPSAPSQSRMTSE RAPSPSSRMGQAPSQSLLPPAQDQPRSPVPSAFSDQSRCLIAQTTPVAGSQSLSSGAV ATTTSSAGDHNGMLSVPAPGVPHSDVGEPPASTGAQQPSALAALQPAKERRSSSSSSS SSSSSSSSSSSSSSSSSSGSSSSDSEGSSLPVQPEVALKRVPSPTPAPKEAVREGRPP EPTPAKRKRRSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSPSPAKP GPQALPKPASPKKPPPGERRSRSPRKPIDSLRDSRSLSYSPVERRRPSPQPSPRDQQS SSSERGSRRGQRGDSRSPSHKRRRETPSPRPMRHRSSRSP';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count2 = 0; count2 < 4500; count2++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaiaasevl vdsaeegsla aaaelaaqkr eqrlrkfrel hlmrnearkl nhqevveedk 61 rlklpanwea kkarlewelk eeekkkecaa rgedyekvkl leisaedaer werkkkrknp 121 dlgfsdyaaa qlrqyhrltk qikpdmetye rlrekhgeef fptsnsllhg thvpsteeid 181 rmvidlekqi ekrdkysrrr pynddadidy inernakfnk kaerfygkyt aeikqnlerg 241 tav';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mafpepkprp pelpqkrlkt ldcgqgavra vrfnvdgnyc ltcgsdktlk lwnplrgtll 61 rtysghgyev ldaagsfdns slcsgggdka vvlwdvasgq vvrkfrghag kvntvqfnee 121 atvilsgsid ssircwdcrs rrpepvqtld eardgvssvk vsdheilags vdgrvrrydl 181 rmgqlfsdyv gspitctcfs rdgqctlvss ldstlrlldk dtgellgeyk ghknqeykld 241 cclserdthv vscsedgkvf fwdlvegala lalpvgsgvv qslayhptep clltamggsv 301 qcwreeayea edgag';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count4 = 0; count4 < 4500; count4++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 massasartp agkrvinqee lrrlmkekqr lstsrkries pfakynrlgq lscalcntpv 61 ksellwqthv lgkqhrekva elkgakeasq gssassaphs vkrkapdadd qdvkrakatl 121 vpqvqpstsa wttnfdkigk efiratpskp sglsllpdye deeeeeeeee gdgerkrgda 181 skplsdaqgk ehsvsssrev tssvlpndff stnppkapii phsgsiekae ihekvverre 241 ntaealpegf fddpevdarv rkvdapkdqm dkewdefqka mrqvntisea ivaeedeegr 301 ldrqigeide qiecyrrvek lrnrqdeikn klkeiltike lqkkeeenad sddegelqdl 361 lsqdwrvkga ll';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maendvdnel ldyeddevet aaggdgaeap akkdvkgsyv sihssgfrdf llkpellrai 61 vdcgfehpse vqhecipqai lgmdvlcqak sgmgktavfv latlqqlepv tgqvsvlvmc 121 htrelafqis keyerfskym pnvkvavffg glsikkdeev lkkncphivv gtpgrilala 181 rnkslnlkhi khfildecdk mleqldmrrd vqeifrmtph ekqvmmfsat lskeirpvcr 241 kfmqdpmeif vddetkltlh glqqyyvklk dneknrklfd lldvlefnqv vifvksvqrc 301 ialaqllveq nfpaiaihrg mpqeerlsry qqfkdfqrri lvatnlfgrg mdiervniaf 361 nydmpedsdt ylhrvaragr fgtkglaitf vsdendakil ndvqdrfevn iselpdeidi 421 ssyieqtr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count6 = 0; count6 < 4500; count6++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maaapplska eylkrylsga dagvdrgses grkrrkkrpk pggaggkgmr ivdddvswta 61 isttklekee eeddgdlpvv aefvderpee vkqmeafrss akwkllgghn edlpsnrhfr 121 hdtpdssprr vrhgtpdpsp rkdrhdtpdp sprrarhdtp dpsplrgarh dsdtspprri 181 rhdssdtspp rrarhdspdp spprrpqhns sgasprrvrh dspdpspprr arhgssdiss 241 prrvhnnspd tsrrtlgssd tqqlrrarhd spdlapnvty slprtksgka perassktsp 301 hwkesgashl sfpknskyey dpdispprkk qakshfgdkk qldskgdcqk atdsdlsspr 361 hkqspghqds dsdlspprnr prhrssdsdl spprrrqrtk ssdsdlsppr rsqppgkkaa 421 hmysgaktgl vltdiqreqq elkeqdqetm afeaefqyae tvfrdksgrk rnlklerleq 481 rrkaekdser delyaqwgkg laqsrqqqqn vedamkemqk plaryidded ldrmlreqer 541 egdpmanfik knkakenknk kvrprysgpa pppnrfniwp gyrwdgvdrs ngfeqkrfar 601 laskkaveel aykwsvedm';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maepdleceq irlkcirkeg fftvppehrl grcrsvkefe klnrigegty givyrardtq 61 tdeivalkkv rmdkekdgip isslreitll lrlrhpnive lkevvvgnhl esiflvmgyc 121 eqdlasllen mptpfseaqv kcivlqvlrg lqylhrnfii hrdlkvsnll mtdkgcvkta 181 dfglaraygv pvkpmtpkvv tlwyrapell lgtttqttsi dmwavgcila ellahrpllp 241 gtseihqidl ivqllgtpse niwpgfsklp lvgqyslrkq pynnlkhkfp wlseaglrll 301 hflfmydpkk ratagdcles syfkekplpc epelmptfph hrnkraapat segqskrckp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count8 = 0; count8 < 4500; count8++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mpvvrkifrr rrgdseseed eqdseevrlk leetrevqnl rkrpngvsav allvgekvqe 61 ettlvddpfq mktggmvdmk klkergkdki seeedlhlgt sfsaetnrrd edadmmkyie 121 telkkrkgiv eheeqkvkpk naedclyelp enirvssakk teemlsnqml sgipevdlgi 181 dakikniist edakarllae qqnkkkdset sfvptnmavn yvqhnrfyhe elnapirrnk 241 eepkarplrv gdtekpeper sppnrkrpan ekatddyhye kfkkmnrry';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 metpgasass lllpaasrpp rkreageaga atskqrvlde eeyieglqtv iqrdffpdve 61 klqaqkeyle aeengdlerm rqiaikfgsa lgkmsreppp pyvtpatfet pevhagtgvv 121 gnkprprgrg ledgeageee ekeplpsldv flsrytsedn asfqeimeva kersrarhaw 181 lyqaeeefek rqkdnlelps aehqaiessq asvetwkyka knslmyypeg vpdeeqlfkk 241 prqvvhkntr flrdpfsqal srcqlqqaaa lnaqhkqgkv gpdgkelipq esprvggfgf 301 vatpspapgv nespmmtwge ventplrveg setpyvdrtp gpafkilepg rrerlglkma 361 neaaaknrak kqealrrvte nlasltpkgl spamspalqr lvsrtaskyt dralrasytp 421 sparsthlkt pasglqtpts tpapgsatrt pltqdpasit dnllqlparr kasdff';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count10 = 0; count10 < 4500; count10++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnilpkkswh vrnkdnvarv rrdeaqaree ekererrvll aqqeartefl rkkarhqnsl 61 peleaaeaga pgsgpvdlfr elleegkgvi rgnkeykeek rqekerqeka lgiltylgqs 121 aaeaqtqppw yqlppgrggp ppgpapdeki ksrldplrem qkhlgkkrqh ggdegsrsrk 181 ekegsekqrp keppsldqlr aerlrreaae rsraeallar vqgralqegq peedetddrr 241 rrynsqfnpq larrprqqdp hlth';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 madvldlhea ggedfamded gdesihklke kakkrkgrgf gseegsrarm redydsveqd 61 gdepgpqrsv egwilfvtgv heeateedih dkfaeygeik nihlnldrrt gylkgytlve 121 yetykeaqaa meglngqdlm gqpisvdwcf vrgppkgkrr ggrrrsrspd rrrr';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count12 = 0; count12 < 4500; count12++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeaaalvwi rgpgfgckav rcasgrctvr dfihrhcqdq nvpvenffvk cngalintsd 61 tvqhgavysl eprlcggkgg fgsmlralga qiekttnrea crdlsgrrlr dvnhekamae 121 wvkqqaerea ekeqkrlerl qrklvepkhc ftspdyqqqc hemaerleds vlkgmqaass 181 kmvsaeisen rkrqwptksq tdrgasagkr rcfwlgmegl etaegsnses sdddseeaps 241 tsgmgfhapk igsngvemaa kfpsgsqrar vvntdhgspe qlqipvtdsg rhiledscae 301 lgeskehmes rmvteteetq ekkaeskepi eeeptgagln kdketeertd gervaevape 361 erenvavakl qesqpgnavi dketidllaf tsvaelellg leklkcelma lglkcggtlq 421 eraarlfsvr glakeqidpa lfakplkgkk k';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mkavkserer gsrrrhrdgd vvlpagvvvk qerlspevap pahrrpdhsg gspspptsep 61 arsghrgnra rgvsrsppkk knkasgrrsk sprskrnrsp hhstvkvkqe redhprrgre 121 drqhrepseq ehrrarnsdr drhrghshqr rtsnerpgsg qgqgrdrdtq nlqaqeeere 181 fynarrrehr qrndvggggs esqelvprpg gnnkekevpa kekpsfelsg alledtntfr 241 gvvikysepp earipkkrwr lypfkndevl pvmyihrqsa yllgrhrria dipidhpscs 301 kqhavfqyrl veytradgtv grrvkpyiid lgsgngtfln nkriepqryy elkekdvlkf 361 gfssreyvll hessdtseid rkddedeeee eevsds';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count14 = 0; count14 < 4500; count14++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 myngiglptp rgsgtngyvq rnlslvrgrr gerpdykgee elrrleaalv krpnpdildh 61 erkrrvelrc leleemmeeq gyeeqqiqek vatfrlmlle kdvnpggkee tpgqrpavte 121 thqlaelnek knerlraafg isdsyvdgss fdpqrrarea kqpapeppkp yslvressss 181 rsptpkqkkk kkkkdrgrrs esssprrerk ksskkkkhrs eseskkrkhr sptpkskrks 241 kdkkrkrsrs ttpapksrra hrstsadsas ssdtsrsrsr saaakthtta lagrspspas 301 grrgegdapf sepgttstqr psspetatkq psspyedkdk dkkeksatrp spspersstg 361 peppaptpll aerhggspqp lattplsqep vnppseaspt rdrsppkspe klpqsssses 421 sppspqptkv srhassspes pkpapapgsh reisssptsk nrshgrakrd kshshtpsrr 481 mgrsrspata krgrsrsrtp tkrghsrsrs pqwrrsrsaq rwgrsrspqr rgrsrspqrp 541 gwsrsrntqr rgrsrsarrg rshsrspatr grsrsrtpar rgrsrsrtpa rrrsrsrtpt 601 rrrsrsrtpa rrgrsrsrtp arrrsrtrsp vrrrsrsrsp arrsgrsrsr tparrgrsrs 661 rtparrgrsr srtparrsgr srsrtparrg rsrsrtprrg rsrsrslvrr grshsrtpqr 721 rgrsgssser knksrtsqrr srsnsspemk ksrissrrsr slssprskak srlslrrsls 781 gsspcpkqks qtpprrsrsg ssqpkaksrt pprrsrssss pppkqksktp srqshssssp 841 hpkvksgtpp rqgsitspqa neqsvtpqrr scfesspdpe lksrtpsrhs csgsspprvk 901 sstpprqsps rssspqpkvk aiisprqrsh sgssspspsr vtsrttprrs rsvspcsnve 961 srllpryshs gssspdtkvk petpprqshs gsispypkvk aqtppgpsls gskspcpqek 1021 skdslvqscp gslslcagvk sstppgesyf gvsslqlkgq sqtspdhrsd tsspevrqsh 1081 sespslqsks qtspkggrsr ssspvtelas rspirqdrge fsaspmlksg mspeqsrfqs 1141 dsssyptvds nsllgqsrle taeskekmal ppqedatasp prqkdkfspf pvqdrpessl 1201 vfkdtlrtpp rersgagssp etkeqnsalp tssqdeelme vvekseepag qilshlssel 1261 kemstsnfes speveerpav sltldqsqsq asleavevps masswggphf spehkelsns 1321 plrensfgsp lefrnsgplg temntgfsse vkedlngpfl nqletdpsld mkeqstrssg 1381 hssselspda vekagmssnq sisspvldav prtpsrerss sasspemkdg lprtpsrrsr 1441 sgsspglrdg sgtpsrhsls gsspgmkdip rtpsrgrsec dsspepkalp qtprprsrsp 1501 sspelnnkcl tpqrersgse ssvdqktvar tplgqrsrsg ssqeldvkps aspqersesd 1561 sspdskaktr tplrqrsrsg sspevdsksr lsprrsrsgs spevkdkpra apraqsgsds 1621 spepkapapr alprrsrsgs sskgrgpspe gssstesspe hppksrtarr gsrsspepkt 1681 ksrtpprrrs srsspeltrk arlsrrsrsa ssspetrsrt pprhrrspsv sspepaeksr 1741 ssrrrrsass prtkttsrrg rspspkprgl qrsrsrsrre ktrttrrrdr sgssqstsrr 1801 rqrsrsrsrv trrrrggsgy hsrsparqes srtssrrrrg rsrtpptsrk rsrsrtspap 1861 wkrsrsrasp athrrsrsrt plisrrrsrs rtspvsrrrs rsrtsvtrrr srsraspvsr 1921 rrsrsrtppv trrrsrsrtp ttrrrsrsrt ppvtrrrsrs rtppvtrrrs rsrtspitrr 1981 rsrsrtspvt rrrsrsrtsp vtrrrsrsrt spvtrrrsrs rtppairrrs rsrtpllprk 2041 rsrsrsplai rrrsrsrtpr targkrsltr sppairrrsa sgsssdrsrs atppatrnhs 2101 gsrtppvaln ssrmscfsrp smsptpldrc rspgmleplg ssrtpmsvlq qaggsmmdgp 2161 gpripdhqrt svpenhaqsr ialaltaisl gtarpppsms aaglaarmsq vpapvplmsl 2221 rtapaanlas ripaasaaam nlasartpai ptavnladsr tpaaaaamnl asprtavaps 2281 avnladprtp tapavnlaga rtpaalaals ltgsgtppta anypsssrtp qapasanlvg 2341 prsahatapv niagsrtaaa lapasltsar mapalsganl tsprvplsay ervsgrtspp 2401 lldrarsrtp psapsqsrmt serapspssr mgqapsqsll ppaqdqprsp vpsafsdqsr 2461 cliaqttpva gsqslssgav atttssagdh ngmlsvpapg vphsdvgepp astgaqqpsa 2521 laalqpaker rsssssssss ssssssssss ssssssgsss sdsegsslpv qpevalkrvp 2581 sptpapkeav regrppeptp akrkrrssss ssssssssss ssssssssss ssssssssss 2641 ssssssssps pakpgpqalp kpaspkkppp gerrsrsprk pidslrdsrs lsyspverrr 2701 pspqpsprdq qssssergsr rgqrgdsrsp shkrrretps prpmrhrssr sp';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mafpepkprp pelpqkrlkt ldcgqgavra vrfnvdgnyc ltcgsdktlk lwnplrgtll 61 rtysghgyev ldaagsfdns slcsgggdka vvlwdvasgq vvrkfrghag kvntvqfnee 121 atvilsgsid ssircwdcrs rrpepvqtld eardgvssvk vsdheilags vdgrvrrydl 181 rmgqlfsdyv gspitctcfs rdgqctlvss ldstlrlldk dtgellgeyk ghknqeykld 241 cclserdthv vscsedgkvf fwdlvegala lalpvgsgvv qslayhptep clltamggsv 301 qcwreeayea edgag';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count16 = 0; count16 < 4500; count16++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mavansspvn pvvffdvsig gqevgrmkie lfadvvpkta enfrqfctge frkdgvpigy 61 kgstfhrvik dfmiqggdfv ngdgtgvasi yrgpfadenf klrhsapgll smansgpstn 121 gcqffitcsk cdwldgkhvv fgkiidgllv mrkienvptg pnnkpklpvv isqcgem';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 malskrelde lkpwiektvk rvlgfseptv vtaalncvgk gmdkkkaadh lkpflddstl 61 rfvdklfeav eegrssrhsk sssdrsrkre lkevfgddse iskessgvkk rriprfeeve 121 eepevipgpp sespgmltkl qikqmmeaat rqieerkkql sfispptpqp ktpsssqper 181 lpigntiqps qaatfmndai ekarkaaelq ariqaqlalk pglignanmv glanlhamgi 241 appkvelkdq tkptplilde qgrtvdatgk eielthrmpt lkaniravkr eqfkqqlkek 301 psedmesntf fdprvsiaps qrqrrtfkfh dkgkfekiaq rlrtkaqlek lqaeisqaar 361 ktgihtstrl aliapkkelk egdipeieww dsyiipngfd lteenpkred yfgitnlveh 421 paqlnppvdn dtpvtlgvyl tkkeqkklrr qtrreaqkel qekvrlglmp ppepkvrisn 481 lmrvlgteav qdptkveahv raqmakrqka heeanaarkl taeqrkvkki kklkedisqg 541 vhisvyrvrn lsnpakkfki eanagqlylt gvvvlhkdvn vvvveggpka qkkfkrlmlh 601 rikwdeqtsn tkgdddeesd eeavkktnkc vlvwegtakd rsfgemkfkq cptenmareh 661 fkkhgaehyw dlalsesvle std';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count18 = 0; count18 < 4500; count18++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 massrasstq atktkapddl vapvvkkphi yygsleeker erlakgesgi lgkdglkagi 61 eagninitsg evfeieehis erqaevlaef errkrarqin vstddsevka clralgepit 121 lfgegpaerr erlrnilsvv gtdalkktkk ddekskkske eyqqtwyheg pnslkvarlw 181 ianyslpram krleearlhk eipettrtsq mqelhkslrs lnnfcsqigd drpisychfs 241 pnskmlatac wsglcklwsv pdcnllhtlr ghntnvgaiv fhpkstvsld pkdvnlasca 301 adgsvklwsl dsdepvadie ghtvrvarvm whpsgrflgt tcydrswrlw dleaqeeilh 361 qeghsmgvyd iafhqdgsla gtggldafgr vwdlrtgrci mfleghlkei yginfspngy 421 hiatgsgdnt ckvwdlrqrr cvytipahqn lvtgvkfepi hgnflltgay dntakiwthp 481 gwsplktlag hegkvmgldi ssdgqliatc sydrtfklwm ae';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msladellad leeaaeeeeg gsygeeeeep aiedvqeetq ldlsgdsvkt iaklwdskmf 61 aeimmkieey iskqakasev mgpveaapey rvivdannlt veienelnii hkfirdkysk 121 rfpeleslvp naldyirtvk elgnsldkck nnenlqqilt natimvvsvt asttqgqqls 181 eeelerleea cdmalelnas khriyeyves rmsfiapnls iiigastaak imgvaggltn 241 lskmpacnim llgaqrktls gfsstsvlph tgyiyhsdiv qslppdlrrk aarlvaakct 301 laarvdsfhe stegkvgyel kdeierkfdk wqepppvkqv kplpapldgq rkkrggrryr 361 kmkerlglte irkqanrmsf geieedayqe dlgfslghlg ksgsgrvrqt qvneatkari 421 sktlqrtlqk qsvvyggkst irdrssgtas svaftplqgl eivnpqaaek kvaeanqkyf 481 ssmaeflkvk geksglmst';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count20 = 0; count20 < 4500; count20++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mteadvnpka ypladahltk klldlvqqsc nykqlrkgan eatktlnrgi sefivmaada 61 epleiilhlp llcedknvpy vfvrskqalg racgvsrpvi acsvtikegs qlkqqiqsiq 121 qsierllv';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mpkrkvtfqg vgdeededei ivpkkklvdp vagsggpgsr fkgkhsldsd eeeddddggs 61 skydilased vegqeaatlp seggvritpf nlqeemeegh fdadgnyfln rdaqirdswl 121 dnidwvkire rppgqrqasd seeedslgqt smsaqalleg llelllpret vagalrrlga 181 rgggkgrkgp gqpsspqrld rlsgladqmv argnlgvyqe trerlamrlk glgcqtlgph 241 nptpppsldm faeelaeeel etptptqrge aesrgdglvd vmweykwent gdaelygpft 301 saqmqtwvse gyfpdgvycr kldppggqfy nskridfdly t';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count22 = 0; count22 < 4500; count22++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mageladkkd rdaspskeer krsrtpdrer drdrdrkssp skdrkrhrsr drrrggsrsr 61 srsrsksaer errhkererd kerdrnkkdr drdkdghrrd kdrkrsslsp grgkdfksrk 121 drdskkdeed ehgdkkpkaq plsleellak kkaeeeaeak pkflskaere aealkrrqqe 181 veerqrmlee erkkrkqfqd lgrkmledpq ererrerrer meretngned eegrqkiree 241 kdkskelhai kerylggikk rrrtrhlndr kfvfewdase dtsidynply kerhqvqllg 301 rgfiagidlk qqkreqsrfy gdlmekrrtl eekeqeearl rklrkkeakq rwddrhwsqk 361 kldemtdrdw rifredysit tkggkipnpi rswkdsslpp hilevidkcg ykeptpiqrq 421 aipiglqnrd iigvaetgsg ktaaflipll vwittlpkid rieesdqgpy aiilaptrel 481 aqqieeetik fgkplgirtv aviggisred qgfrlrmgce iviatpgrli dvlenrylvl 541 srctyvvlde adrmidmgfe pdvqkilehm pvsnqkpdtd eaedpekmla nfesgkhkyr 601 qtvmftatmp paverlarsy lrrpavvyig sagkpherve qkvflmsese krkkllaile 661 qgfdppiiif vnqkkgcdvl akslekmgyn actlhggkgq eqrefalsnl kagakdilva 721 tdvagrgidi qdvsmvvnyd makniedyih rigrtgragk sgvaitfltk edsavfyelk 781 qailespvss cppelanhpd aqhkpgtilt kkrreetifa';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnkkkkpflg mpaplgyvpg lgrgatgftt rsdigparda ndpvddrhap pgkrtvgdqm 61 kknqaadddd edlndtnyde fngyagslfs sgpyekddee adaiyaaldk rmderrkerr 121 eqrekeeiek yrmerpkiqq qfsdlkrkla evteeewlsi pevgdarnkr qrnpryeklt 181 pvpdsffakh lqtgenhtsv dprqtqfggl ntpypgglnt pypggmtpgl mtpgtgeldm 241 rkigqarntl mdmrlsqvsd svsgqtvvdp kgyltdlnsm ipthggdind ikkarlllks 301 vretnphhpp awiasarlee vtgklqvarn limkgtemcp ksedvwleaa rlqpgdtaka 361 vvaqavrhlp qsvriyiraa eletdirakk rvlrkalehv pnsvrlwkaa veleepedar 421 imlsravecc ptsvelwlal arletyenar kvlnkareni ptdrhiwita akleeangnt 481 qmvekiidra itslrangve inreqwiqda eecdragsva tcqavmravi gigieeedrk 541 htwmedadsc vahnalecar aiyayalqvf pskksvwlra ayfeknhgtr esleallqra 601 vahcpkaevl wlmgakskwl agdvpaarsi lalafqanpn seeiwlaavk lesendeyer 661 arrllakars saptarvfmk svklewvqdn iraaqdlcee alrhyedfpk lwmmkgqiee 721 qkemmekare aynqglkkcp hstplwllls rleekigqlt raraileksr lknpknpglw 781 lesvrleyra glkniantlm akalqecpns gilwseaifl earpqrrtks vdalkkcehd 841 phvllavakl fwsqrkitka rewfhrtvki dsdlgdawaf fykfelqhgt eeqqeevrkr 901 cesaeprhge lwcavskdia nwqkkigdil rlvagriknt f';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count24 = 0; count24 < 4500; count24++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 magvfpyrgp gnpvpgplap lpdymseekl qekarkwqql qakryaekrk fgfvdaqked 61 mppehvrkii rdhgdmtnrk frhdkrvylg alkymphavl kllenmpmpw eqirdvpvly 121 hitgaisfvn eipwviepvy isqwgsmwim mrrekrdrrh fkrmrfppfd deeppldyad 181 nildveplea iqleldpeed apvldwfydh qplrdsrkyv ngstyqrwqf tlpmmstlyr 241 lanqlltdlv ddnyfylfdl kafftskaln maipggpkfe plvrdinlqd edwnefndin 301 kiiirqpirt eykiafpyly nnlphhvhlt wyhtpnvvfi ktedpdlpaf yfdplinpis 361 hrhsvksqep lpdddeefel pefvepflkd tplytdntan giallwaprp fnlrsgrtrr 421 aldiplvknw yrehcpagqp vkvrvsyqkl lkyyvlnalk hrppkaqkkr ylfrsfkatk 481 ffqstkldwv evglqvcrqg ynmlnllihr knlnylhldy nfnlkpvktl ttkerkksrf 541 gnafhlcrev lrltklvvds hvqyrlgnvd afqladglqy ifahvgqltg myrykyklmr 601 qirmckdlkh liyyrfntgp vgkgpgcgfw aagwrvwlff mrgitpller wlgnllarqf 661 egrhskgvak tvtkqrvesh fdlelraavm hdildmmpeg ikqnkartil qhlseawrcw 721 kanipwkvpg lptpienmil ryvkakadww tntahynrer irrgatvdkt vckknlgrlt 781 rlylkaeqer qhnylkdgpy itaeeavavy tttvhwlesr rfspipfppl sykhdtklli 841 lalerlkeay svksrlnqsq reelglieqa ydnphealsr ikrhlltqra fkevgiefmd 901 lyshlvpvyd veplekitda yldqylwyea dkrrlfppwi kpadtepppl lvykwcqgin 961 nlqdvwetse gecnvmlesr fekmyekidl tllnrllrli vdhniadymt aknnvvinyk 1021 dmnhtnsygi irglqfasfi vqyyglvmdl lvlglhrase magppqmpnd flsfqdiate 1081 aahpirlfcr yidrihiffr ftadeardli qryltehpdp nnenivgynn kkcwprdarm 1141 rlmkhdvnlg ravfwdiknr lprsvttvqw ensfvsvysk dnpnllfnmc gfecrilpkc 1201 rtsyeefthk dgvwnlqnev tkertaqcfl rvddesmqrf hnrvrqilma sgsttftkiv 1261 nkwntaligl mtyfreavvn tqelldllvk cenkiqtrik iglnskmpsr fppvvfytpk 1321 elgglgmlsm ghvlipqsdl rwskqtdvgi thfrsgmshe edqlipnlyr yiqpwesefi 1381 dsqrvwaeya lkrqeaiaqn rrltledled swdrgiprin tlfqkdrhtl aydkgwrvrt 1441 dfkqyqvlkq npfwwthqrh dgklwnlnny rtdmiqalgg vegilehtlf kgtyfptweg 1501 lfwekasgfe esmkwkkltn aqrsglnqip nrrftlwwsp tinranvyvg fqvqldltgi 1561 fmhgkiptlk isliqifrah lwqkihesiv mdlcqvfdqe ldaleietvq ketihprksy 1621 kmnsscadil lfasykwnvs rpslladskd vmdstttqky widiqlrwgd ydshdierya 1681 rakfldyttd nmsiypsptg vliaidlayn lhsaygnwfp gskpliqqam akimkanpal 1741 yvlrerirkg lqlyssepte pylssqnyge lfsnqiiwfv ddtnvyrvti hktfegnltt 1801 kpingaifif nprtgqlflk iihtsvwagq krlgqlakwk taeevaalir slpveeqpkq 1861 iivtrkgmld plevhlldfp nivikgselq lpfqaclkve kfgdlilkat epqmvlfnly 1921 ddwlktissy tafsrlilil ralhvnndra kvilkpdktt itephhiwpt ltdeewikve 1981 vqlkdlilad ygkknnvnva sltqseirdi ilgmeisaps qqrqqiaeie kqtkeqsqlt 2041 atqtrtvnkh gdeiitstts nyetqtfssk tewrvraisa anlhlrtnhi yvssddiket 2101 gytyilpknv lkkficisdl raqiagylyg vsppdnpqvk eircivmvpq wgthqtvhlp 2161 gqlpqheylk emeplgwiht qpnespqlsp qdvtthakim adnpswdgek tiiitcsftp 2221 gsctltaykl tpsgyewgrq ntdkgnnpkg ylpshyervq mllsdrflgf fmvpaqsswn 2281 ynfmgvrhdp nmkyelqlan pkefyhevhr pshflnfall qegevysadr edlya';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

if (unused\_variable == 5) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msymlphlhn gwqvdqails eedrvvvirf ghdwdptcmk mdevlysiae kvknfaviyl 61 vditevpdfn kmyelydpct vmfffrnkhi midlgtgnnn kinwamedkq emvdiietvy 121 rgarkgrglv vspkdystky ry';

protein();

}

cursor1x = cursor1x - 1.23758547e+26;

}

cursor1x = cursor1x + (6.187927353e+28 \* 2) \* 4500;

cursor1y = cursor1y - 2.475170941e+26 \* 1;

}

}

}

function alveolocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count2 = 0; count2 < 1; count2++) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwsggsgkar gweaaaggrs spgrlsrrri msmspkhttp fsvsdilspl eesykkvgme 61 ggglgaplaa yrqgqaappt aamqqhavgh hgavtaayhm taagvpqlsh savggycngn 121 lgnmselppy qdtmrnsasg pgwyganpdp rfpaisrfmg pasgmnmsgm gglgslgdvs 181 knmaplpsap rrkrrvlfsq aqvyelerrf kqqkylsape rehlasmihl tptqvkiwfq 241 nhrykmkrqa kdkaaqqqlq qdsggggggg gtgcpqqqqa qqqsprrvav pvlvkdgkpc 301 qagapapgaa slqghaqqqa qhqaqaaqaa aaaisvgsgg aglgahpghq pgsagqspdl 361 ahhaaspaal qgqvsslshl nssgsdygtm scstllygrt w';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count4 = 0; count4 < 1; count4++) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavlrqlall lwknytlqkr kvlvtvlelf lpllfsgili wlrlkiqsen vpnatiypgq 61 siqelplfft fpppgdtwel ayipshsdaa ktvtetvrra lvinmrvrgf psekdfedyi 121 rydncsssvl aavvfehpfn hskeplplav kyhlrfsytr rnymwtqtgs fflketegwh 181 ttslfplfpn pgpreptspd ggepgyireg flavqhavdr aimeyhadaa trqlfqrltv 241 tikrfpyppf iadpflvaiq yqlpllllls ftytaltiar avvqekerrl keymrmmgls 301 swlhwsawfl lfflflliaa sfmtllfcvk vkpnvavlsr sdpslvlafl lcfaistisf 361 sfmvstffsk anmaaafggf lyfftyipyf fvaprynwmt lsqklcscll snvamamgaq 421 ligkfeakgm giqwrdllsp vnvdddfcfg qvlgmlllds vlyglvtwym eavfpgqfgv 481 pqpwyffimp sywcgkprav agkeeedsdp ekalrneyfe aepedlvagi kikhlskvfr 541 vgnkdraavr dlnlnlyegq itvllghnga gktttlsmlt glfpptsgra yisgyeisqd 601 mvqirkslgl cpqhdilfdn ltvaehlyfy aqlkglsrqk cpeevkqmlh iigledkwns 661 rsrflsggmr rklsigiali agskvlilde ptsgmdaisr raiwdllqrq ksdrtivltt 721 hfmdeadllg driaimakge lqccgsslfl kqkygagyhm tlvkephcnp edisqlvhhh 781 vpnatlessa gaelsfilpr esthrfeglf aklekkqkel giasfgasit tmeevflrvg 841 klvdssmdiq aiqlpalqyq herrasdwav dsnlcgamdp sdgigaliee ertavklntg 901 lalhcqqfwa mflkkaaysw rewkmvaaqv lvpltcvtla llainyssel fddpmlrltl 961 geygrtvvpf svpgtsqlgq qlsehlkdal qaegqeprev lgdleeflif rasvegggfn 1021 erclvaasfr dvgertvvna lfnnqayhsp atalavvdnl lfkllcgpha sivvsnfpqp 1081 rsalqaakdq fnegrkgfdi alnllfamaf lastfsilav seravqakhv qfvsgvhvas 1141 fwlsallwdl isflipslll lvvfkafdvr aftrdghmad tllllllygw aiiplmylmn 1201 ffflgaatay trltifnils giatflmvti mripavklee lsktldhvfl vlpnhclgma 1261 vssfyenyet rryctsseva ahyckkyniq yqenfyawsa pgvgrfvasm aasgcaylil 1321 lflietnllq rlrgilcalr rrrtltelyt rmpvlpedqd vadertrila pspdsllhtp 1381 liikelskvy eqrvpllavd rlslavqkge cfgllgfnga gktttfkmlt geesltsgda 1441 fvgghrissd vgkvrqrigy cpqfdalldh mtgremlvmy arlrgiperh igacventlr 1501 glllephank lvrtysggnk rklstgiali gepaviflde pstgmdpvar rllwdtvara 1561 resgkaiiit shsmeeceal ctrlaimvqg qfkclgspqh lkskfgsgys lrakvqsegq 1621 qealeefkaf vdltfpgsvl edehqgmvhy hlpgrdlswa kvfgilekak ekygvddysv 1681 sqisleqvfl sfahlqppta eegr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count6 = 0; count6 < 1; count6++) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mavlrqlall lwknytlqkr kvlvtvlelf lpllfsgili wlrlkiqsen vpnatiypgq 61 siqelplfft fpppgdtwel ayipshsdaa ktvtetvrra lvinmrvrgf psekdfedyi 121 rydncsssvl aavvfehpfn hskeplplav kyhlrfsytr rnymwtqtgs fflketegwh 181 ttslfplfpn pgpreptspd ggepgyireg flavqhavdr aimeyhadaa trqlfqrltv 241 tikrfpyppf iadpflvaiq yqlpllllls ftytaltiar avvqekerrl keymrmmgls 301 swlhwsawfl lfflflliaa sfmtllfcvk vkpnvavlsr sdpslvlafl lcfaistisf 361 sfmvstffsk anmaaafggf lyfftyipyf fvaprynwmt lsqklcscll snvamamgaq 421 ligkfeakgm giqwrdllsp vnvdddfcfg qvlgmlllds vlyglvtwym eavfpgqfgv 481 pqpwyffimp sywcgkprav agkeeedsdp ekalrneyfe aepedlvagi kikhlskvfr 541 vgnkdraavr dlnlnlyegq itvllghnga gktttlsmlt glfpptsgra yisgyeisqd 601 mvqirkslgl cpqhdilfdn ltvaehlyfy aqlkglsrqk cpeevkqmlh iigledkwns 661 rsrflsggmr rklsigiali agskvlilde ptsgmdaisr raiwdllqrq ksdrtivltt 721 hfmdeadllg driaimakge lqccgsslfl kqkygagyhm tlvkephcnp edisqlvhhh 781 vpnatlessa gaelsfilpr esthrfeglf aklekkqkel giasfgasit tmeevflrvg 841 klvdssmdiq aiqlpalqyq herrasdwav dsnlcgamdp sdgigaliee ertavklntg 901 lalhcqqfwa mflkkaaysw rewkmvaaqv lvpltcvtla llainyssel fddpmlrltl 961 geygrtvvpf svpgtsqlgq qlsehlkdal qaegqeprev lgdleeflif rasvegggfn 1021 erclvaasfr dvgertvvna lfnnqayhsp atalavvdnl lfkllcgpha sivvsnfpqp 1081 rsalqaakdq fnegrkgfdi alnllfamaf lastfsilav seravqakhv qfvsgvhvas 1141 fwlsallwdl isflipslll lvvfkafdvr aftrdghmad tllllllygw aiiplmylmn 1201 ffflgaatay trltifnils giatflmvti mripavklee lsktldhvfl vlpnhclgma 1261 vssfyenyet rryctsseva ahyckkyniq yqenfyawsa pgvgrfvasm aasgcaylil 1321 lflietnllq rlrgilcalr rrrtltelyt rmpvlpedqd vadertrila pspdsllhtp 1381 liikelskvy eqrvpllavd rlslavqkge cfgllgfnga gktttfkmlt geesltsgda 1441 fvgghrissd vgkvrqrigy cpqfdalldh mtgremlvmy arlrgiperh igacventlr 1501 glllephank lvrtysggnk rklstgiali gepaviflde pstgmdpvar rllwdtvara 1561 resgkaiiit shsmeeceal ctrlaimvqg qfkclgspqh lkskfgsgys lrakvqsegq 1621 qealeefkaf vdltfpgsvl edehqgmvhy hlpgrdlswa kvfgilekak ekygvddysv 1681 sqisleqvfl sfahlqppta eegr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count8 = 0; count8 < 1; count8++) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwsggsgkar gweaaaggrs spgrlsrrri msmspkhttp fsvsdilspl eesykkvgme 61 ggglgaplaa yrqgqaappt aamqqhavgh hgavtaayhm taagvpqlsh savggycngn 121 lgnmselppy qdtmrnsasg pgwyganpdp rfpaisrfmg pasgmnmsgm gglgslgdvs 181 knmaplpsap rrkrrvlfsq aqvyelerrf kqqkylsape rehlasmihl tptqvkiwfq 241 nhrykmkrqa kdkaaqqqlq qdsggggggg gtgcpqqqqa qqqsprrvav pvlvkdgkpc 301 qagapapgaa slqghaqqqa qhqaqaaqaa aaaisvgsgg aglgahpghq pgsagqspdl 361 ahhaaspaal qgqvsslshl nssgsdygtm scstllygrt w';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['alveolocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function bronchocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count10 = 0; count10 < 1; count10++) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msqesdnnkr lvalvpmpsd ppfntrrayt sedeawksyl enpltaatka mmsingdeds 61 aaalgllydy ykvprdkrll svskasdsqe dqekrnclgt seaqsnlsgg enrvqvlktv 121 pvnlslnqdh lenskreqys isfpessaii pvsgitvvka edftpvfmap pvhyprgdge 181 eqrvvifeqt qydvpslath saylkddqrs tpdstysesf kdaatekfrs asvgaeeymy 241 dqtssgtfqy tleatkslrq kqgegpmtyl nkgqfyaitl setgdnkcfr hpiskvrsvv 301 mvvfsedknr deqlkywkyw hsrqhtakqr vldiadykes fntignieei aynavsftwd 361 vneeakifit vnclstdfss qkgvkglplm iqidtysynn rsnkpihray cqikvfcdkg 421 aerkirdeer kqnrkkgkgq asqtqcnsss dgklaaiplq kksdityfkt mpdlhsqpvl 481 fipdvhfanl qrtgqvyynt ddereggsvl vkrmfrpmee efgpvpskqm keegtkrvll 541 yvrketddvf dalmlksptv kglmeaisek yglpvekiak lykkskkgil vnmddniieh 601 ysnedtfiln mesmvegfkv tlmei';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count12 = 0; count12 < 1; count12++) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msqesdnnkr lvalvpmpsd ppfntrrayt sedeawksyl enpltaatka mmsingdeds 61 aaalgllydy ykvprdkrll svskasdsqe dqekrnclgt seaqsnlsgg enrvqvlktv 121 pvnlslnqdh lenskreqys isfpessaii pvsgitvvka edftpvfmap pvhyprgdge 181 eqrvvifeqt qydvpslath saylkddqrs tpdstysesf kdaatekfrs asvgaeeymy 241 dqtssgtfqy tleatkslrq kqgegpmtyl nkgqfyaitl setgdnkcfr hpiskvrsvv 301 mvvfsedknr deqlkywkyw hsrqhtakqr vldiadykes fntignieei aynavsftwd 361 vneeakifit vnclstdfss qkgvkglplm iqidtysynn rsnkpihray cqikvfcdkg 421 aerkirdeer kqnrkkgkgq asqtqcnsss dgklaaiplq kksdityfkt mpdlhsqpvl 481 fipdvhfanl qrtgqvyynt ddereggsvl vkrmfrpmee efgpvpskqm keegtkrvll 541 yvrketddvf dalmlksptv kglmeaisek yglpvekiak lykkskkgil vnmddniieh 601 ysnedtfiln mesmvegfkv tlmei';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count14 = 0; count14 < 1; count14++) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwsggsgkar gweaaaggrs spgrlsrrri msmspkhttp fsvsdilspl eesykkvgme 61 ggglgaplaa yrqgqaappt aamqqhavgh hgavtaayhm taagvpqlsh savggycngn 121 lgnmselppy qdtmrnsasg pgwyganpdp rfpaisrfmg pasgmnmsgm gglgslgdvs 181 knmaplpsap rrkrrvlfsq aqvyelerrf kqqkylsape rehlasmihl tptqvkiwfq 241 nhrykmkrqa kdkaaqqqlq qdsggggggg gtgcpqqqqa qqqsprrvav pvlvkdgkpc 301 qagapapgaa slqghaqqqa qhqaqaaqaa aaaisvgsgg aglgahpghq pgsagqspdl 361 ahhaaspaal qgqvsslshl nssgsdygtm scstllygrt w';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count16 = 0; count16 < 1; count16++) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwsggsgkar gweaaaggrs spgrlsrrri msmspkhttp fsvsdilspl eesykkvgme 61 ggglgaplaa yrqgqaappt aamqqhavgh hgavtaayhm taagvpqlsh savggycngn 121 lgnmselppy qdtmrnsasg pgwyganpdp rfpaisrfmg pasgmnmsgm gglgslgdvs 181 knmaplpsap rrkrrvlfsq aqvyelerrf kqqkylsape rehlasmihl tptqvkiwfq 241 nhrykmkrqa kdkaaqqqlq qdsggggggg gtgcpqqqqa qqqsprrvav pvlvkdgkpc 301 qagapapgaa slqghaqqqa qhqaqaaqaa aaaisvgsgg aglgahpghq pgsagqspdl 361 ahhaaspaal qgqvsslshl nssgsdygtm scstllygrt w';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['bronchocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function bulbocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count18 = 0; count18 < 1; count18++) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgptsgpsll llllthlpla lgspmysiit pnilrlesee tmvleahdaq gdvpvtvtvh 61 dfpgkklvls sektvltpat nhmgnvtfti panrefksek grnkfvtvqa tfgtqvvekv 121 vlvslqsgyl fiqtdktiyt pgstvlyrif tvnhkllpvg rtvmvnienp egipvkqdsl 181 ssqnqlgvlp lswdipelvn mgqwkirayy enspqqvfst efevkeyvlp sfevivepte 241 kfyyiynekg levtitarfl ygkkvegtaf vifgiqdgeq rislpeslkr ipiedgsgev 301 vlsrkvlldg vqnpraedlv gkslyvsatv ilhsgsdmvq aersgipivt spyqihftkt 361 pkyfkpgmpf dlmvfvtnpd gspayrvpva vqgedtvqsl tqgdgvakls inthpsqkpl 421 sitvrtkkqe lseaeqatrt mqalpystvg nsnnylhlsv lrtelrpget lnvnfllrmd 481 raheakiryy tylimnkgrl lkagrqvrep gqdlvvlpls ittdfipsfr lvayytliga 541 sgqrevvads vwvdvkdscv gslvvksgqs edrqpvpgqq mtlkiegdhg arvvlvavdk 601 gvfvlnkknk ltqskiwdvv ekadigctpg sgkdyagvfs dagltftsss gqqtaqrael 661 qcpqpaarrr rsvqltekrm dkvgkypkel rkccedgmre npmrfscqrr trfislgeac 721 kkvfldccny itelrrqhar ashlglarsn ldediiaeen ivsrsefpes wlwnvedlke 781 ppkngistkl mniflkdsit tweilavsms dkkgicvadp fevtvmqdff idlrlpysvv 841 rneqveirav lynyrqnqel kvrvellhnp afcslattkr rhqqtvtipp ksslsvpyvi 901 vplktglqev evkaavyhhf isdgvrkslk vvpegirmnk tvavrtldpe rlgregvqke 961 dippadlsdq vpdtesetri llqgtpvaqm tedavdaerl khlivtpsgc geqnmigmtp 1021 tviavhylde teqwekfgle krqgalelik kgytqqlafr qpssafaafv krapstwlta 1081 yvvkvfslav nliaidsqvl cgavkwlile kqkpdgvfqe dapvihqemi gglrnnnekd 1141 maltafvlis lqeakdicee qvnslpgsit kagdfleany mnlqrsytva iagyalaqmg 1201 rlkgpllnkf lttakdknrw edpgkqlynv eatsyallal lqlkdfdfvp pvvrwlneqr 1261 yygggygstq atfmvfqala qyqkdapdhq elnldvslql psrsskithr ihwesasllr 1321 seetkenegf tvtaegkgqg tlsvvtmyha kakdqltcnk fdlkvtikpa petekrpqda 1381 kntmileict ryrgdqdatm sildismmtg fapdtddlkq langvdryis kyeldkafsd 1441 rntliiyldk vshseddcla fkvhqyfnve liqpgavkvy ayynleesct rfyhpekedg 1501 klnklcrdel crcaeencfi qksddkvtle erldkacepg vdyvyktrlv kvqlsndfde 1561 yimaieqtik sgsdevqvgq qrtfispikc realkleekk hylmwglssd fwgekpnlsy 1621 iigkdtwveh wpeedecqde enqkqcqdlg aftesmvvfg cpn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count20 = 0; count20 < 1; count20++) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgptsgpsll llllthlpla lgspmysiit pnilrlesee tmvleahdaq gdvpvtvtvh 61 dfpgkklvls sektvltpat nhmgnvtfti panrefksek grnkfvtvqa tfgtqvvekv 121 vlvslqsgyl fiqtdktiyt pgstvlyrif tvnhkllpvg rtvmvnienp egipvkqdsl 181 ssqnqlgvlp lswdipelvn mgqwkirayy enspqqvfst efevkeyvlp sfevivepte 241 kfyyiynekg levtitarfl ygkkvegtaf vifgiqdgeq rislpeslkr ipiedgsgev 301 vlsrkvlldg vqnpraedlv gkslyvsatv ilhsgsdmvq aersgipivt spyqihftkt 361 pkyfkpgmpf dlmvfvtnpd gspayrvpva vqgedtvqsl tqgdgvakls inthpsqkpl 421 sitvrtkkqe lseaeqatrt mqalpystvg nsnnylhlsv lrtelrpget lnvnfllrmd 481 raheakiryy tylimnkgrl lkagrqvrep gqdlvvlpls ittdfipsfr lvayytliga 541 sgqrevvads vwvdvkdscv gslvvksgqs edrqpvpgqq mtlkiegdhg arvvlvavdk 601 gvfvlnkknk ltqskiwdvv ekadigctpg sgkdyagvfs dagltftsss gqqtaqrael 661 qcpqpaarrr rsvqltekrm dkvgkypkel rkccedgmre npmrfscqrr trfislgeac 721 kkvfldccny itelrrqhar ashlglarsn ldediiaeen ivsrsefpes wlwnvedlke 781 ppkngistkl mniflkdsit tweilavsms dkkgicvadp fevtvmqdff idlrlpysvv 841 rneqveirav lynyrqnqel kvrvellhnp afcslattkr rhqqtvtipp ksslsvpyvi 901 vplktglqev evkaavyhhf isdgvrkslk vvpegirmnk tvavrtldpe rlgregvqke 961 dippadlsdq vpdtesetri llqgtpvaqm tedavdaerl khlivtpsgc geqnmigmtp 1021 tviavhylde teqwekfgle krqgalelik kgytqqlafr qpssafaafv krapstwlta 1081 yvvkvfslav nliaidsqvl cgavkwlile kqkpdgvfqe dapvihqemi gglrnnnekd 1141 maltafvlis lqeakdicee qvnslpgsit kagdfleany mnlqrsytva iagyalaqmg 1201 rlkgpllnkf lttakdknrw edpgkqlynv eatsyallal lqlkdfdfvp pvvrwlneqr 1261 yygggygstq atfmvfqala qyqkdapdhq elnldvslql psrsskithr ihwesasllr 1321 seetkenegf tvtaegkgqg tlsvvtmyha kakdqltcnk fdlkvtikpa petekrpqda 1381 kntmileict ryrgdqdatm sildismmtg fapdtddlkq langvdryis kyeldkafsd 1441 rntliiyldk vshseddcla fkvhqyfnve liqpgavkvy ayynleesct rfyhpekedg 1501 klnklcrdel crcaeencfi qksddkvtle erldkacepg vdyvyktrlv kvqlsndfde 1561 yimaieqtik sgsdevqvgq qrtfispikc realkleekk hylmwglssd fwgekpnlsy 1621 iigkdtwveh wpeedecqde enqkqcqdlg aftesmvvfg cpn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count22 = 0; count22 < 1; count22++) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mgptsgpsll llllthlpla lgspmysiit pnilrlesee tmvleahdaq gdvpvtvtvh 61 dfpgkklvls sektvltpat nhmgnvtfti panrefksek grnkfvtvqa tfgtqvvekv 121 vlvslqsgyl fiqtdktiyt pgstvlyrif tvnhkllpvg rtvmvnienp egipvkqdsl 181 ssqnqlgvlp lswdipelvn mgqwkirayy enspqqvfst efevkeyvlp sfevivepte 241 kfyyiynekg levtitarfl ygkkvegtaf vifgiqdgeq rislpeslkr ipiedgsgev 301 vlsrkvlldg vqnpraedlv gkslyvsatv ilhsgsdmvq aersgipivt spyqihftkt 361 pkyfkpgmpf dlmvfvtnpd gspayrvpva vqgedtvqsl tqgdgvakls inthpsqkpl 421 sitvrtkkqe lseaeqatrt mqalpystvg nsnnylhlsv lrtelrpget lnvnfllrmd 481 raheakiryy tylimnkgrl lkagrqvrep gqdlvvlpls ittdfipsfr lvayytliga 541 sgqrevvads vwvdvkdscv gslvvksgqs edrqpvpgqq mtlkiegdhg arvvlvavdk 601 gvfvlnkknk ltqskiwdvv ekadigctpg sgkdyagvfs dagltftsss gqqtaqrael 661 qcpqpaarrr rsvqltekrm dkvgkypkel rkccedgmre npmrfscqrr trfislgeac 721 kkvfldccny itelrrqhar ashlglarsn ldediiaeen ivsrsefpes wlwnvedlke 781 ppkngistkl mniflkdsit tweilavsms dkkgicvadp fevtvmqdff idlrlpysvv 841 rneqveirav lynyrqnqel kvrvellhnp afcslattkr rhqqtvtipp ksslsvpyvi 901 vplktglqev evkaavyhhf isdgvrkslk vvpegirmnk tvavrtldpe rlgregvqke 961 dippadlsdq vpdtesetri llqgtpvaqm tedavdaerl khlivtpsgc geqnmigmtp 1021 tviavhylde teqwekfgle krqgalelik kgytqqlafr qpssafaafv krapstwlta 1081 yvvkvfslav nliaidsqvl cgavkwlile kqkpdgvfqe dapvihqemi gglrnnnekd 1141 maltafvlis lqeakdicee qvnslpgsit kagdfleany mnlqrsytva iagyalaqmg 1201 rlkgpllnkf lttakdknrw edpgkqlynv eatsyallal lqlkdfdfvp pvvrwlneqr 1261 yygggygstq atfmvfqala qyqkdapdhq elnldvslql psrsskithr ihwesasllr 1321 seetkenegf tvtaegkgqg tlsvvtmyha kakdqltcnk fdlkvtikpa petekrpqda 1381 kntmileict ryrgdqdatm sildismmtg fapdtddlkq langvdryis kyeldkafsd 1441 rntliiyldk vshseddcla fkvhqyfnve liqpgavkvy ayynleesct rfyhpekedg 1501 klnklcrdel crcaeencfi qksddkvtle erldkacepg vdyvyktrlv kvqlsndfde 1561 yimaieqtik sgsdevqvgq qrtfispikc realkleekk hylmwglssd fwgekpnlsy 1621 iigkdtwveh wpeedecqde enqkqcqdlg aftesmvvfg cpn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['bulbocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function endotheliocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count24 = 0; count24 < 1; count24++) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 metkekpkpt pdylmqlmnd kklmsslpnf cgifnhlerl ldeeisrvrk dmyndtlngs 61 tekrsaelpd avgpivqlqe klyvpvkeyp dfnfvgrilg prgltakqle aetgckimvr 121 gkgsmrdkkk eeqnrgkpnw ehlnedlhvl itvedaqnra eiklkravee vkkllvpaae 181 gedslkkmql melailngty rdanikspal afslaataqa apriitgpap vlppaalrtp 241 tpagptimpl irqiqtavmp ngtphptaai vppgpeagli ytpyeypytl apatsileyp 301 iepsgvlgav atkvrrhdmr vhpyqrivta draatgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count26 = 0; count26 < 1; count26++) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 metkekpkpt pdylmqlmnd kklmsslpnf cgifnhlerl ldeeisrvrk dmyndtlngs 61 tekrsaelpd avgpivqlqe klyvpvkeyp dfnfvgrilg prgltakqle aetgckimvr 121 gkgsmrdkkk eeqnrgkpnw ehlnedlhvl itvedaqnra eiklkravee vkkllvpaae 181 gedslkkmql melailngty rdanikspal afslaataqa apriitgpap vlppaalrtp 241 tpagptimpl irqiqtavmp ngtphptaai vppgpeagli ytpyeypytl apatsileyp 301 iepsgvlgav atkvrrhdmr vhpyqrivta draatgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count28 = 0; count28 < 1; count28++) {

for (var count27 = 0; count27 < 4500; count27++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 metkekpkpt pdylmqlmnd kklmsslpnf cgifnhlerl ldeeisrvrk dmyndtlngs 61 tekrsaelpd avgpivqlqe klyvpvkeyp dfnfvgrilg prgltakqle aetgckimvr 121 gkgsmrdkkk eeqnrgkpnw ehlnedlhvl itvedaqnra eiklkravee vkkllvpaae 181 gedslkkmql melailngty rdanikspal afslaataqa apriitgpap vlppaalrtp 241 tpagptimpl irqiqtavmp ngtphptaai vppgpeagli ytpyeypytl apatsileyp 301 iepsgvlgav atkvrrhdmr vhpyqrivta draatgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count30 = 0; count30 < 1; count30++) {

for (var count29 = 0; count29 < 4500; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 metkekpkpt pdylmqlmnd kklmsslpnf cgifnhlerl ldeeisrvrk dmyndtlngs 61 tekrsaelpd avgpivqlqe klyvpvkeyp dfnfvgrilg prgltakqle aetgckimvr 121 gkgsmrdkkk eeqnrgkpnw ehlnedlhvl itvedaqnra eiklkravee vkkllvpaae 181 gedslkkmql melailngty rdanikspal afslaataqa apriitgpap vlppaalrtp 241 tpagptimpl irqiqtavmp ngtphptaai vppgpeagli ytpyeypytl apatsileyp 301 iepsgvlgav atkvrrhdmr vhpyqrivta draatgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['endotheliocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function ionocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count32 = 0; count32 < 1; count32++) {

for (var count31 = 0; count31 < 4500; count31++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mssfdlpaps pprcspqfps igqeppemnl yyenffhpqg vpspqrpsfe gggeygatpn 61 pylwfngptm tpppylpgpn aspflpqayg vqrpllpsvs glggsdlgwl pipsqeelmk 121 lvrppysysa liamaihgap dkrltlsqiy qyvadnfpfy nkskagwqns irhnlslndc 181 fkkvprdedd pgkgnywtld pncekmfdng nfrrkrkrks dvssstasla lektesslpv 241 dspkttepqd ildgaspggt tsspekrpsp ppsgapclns flssmtayvs ggsptshplv 301 tpglspepsd ktgqnsltfn sfspltnlsn hsgggdwanp mptnmlsygg svlsqfsphf 361 ynsvntsgvl ypregtev';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count34 = 0; count34 < 1; count34++) {

for (var count33 = 0; count33 < 4500; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtsfvpqsls pqfhsmgqes qefslygdnf ysaqhvpspq qtlpsaydfg eyagqtsnpy 61 lwfngpglsp apclttgpqh ygmakqyvga sgiggsegaf swfslpsqed lmklvrppys 121 ysaliamaih gapnrrltls qiyqyvadnf pfynkskasw qnsirhnlsl ndcfmkvprd 181 dsdpgkgnyw tldpncekmf dngnfrrkrk rksdslaeee gkgysgsdsa lsspknpsds 241 sergnspist dqapclnsfl nqmgdvasgs reallpspla vplsqrsspt gvygsyspna 301 tmpqwetqip qssisstpyk dgysdsmlnp yssqlypvlg ssdllypreg sev';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count36 = 0; count36 < 1; count36++) {

for (var count35 = 0; count35 < 4500; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtsyesqgqs ptrcgpqfls lgqeppelsl ysdsyyppps lpspqrtnps syelgdyaas 61 spnpylwfns pgmnsapylg gtpgpagpsf vpqhygmqrp ylgpgppggp ggelswfsmp 121 sqedlmklvr ppysysalia maihgaperr ltlsqiyqyv adnfpfynks kagwqnsirh 181 nlslndcfkk vprdeddpgk gnywtldpnc ekmfdngnfr rkrkrksdsl peksssggne 241 sgdsngrgsp ksqsidists pekgpspast gpspclsnfl temsgvaags ldmeadplsr 301 pftlslpvdg aqrasqttgf stftpsttvs dwasplpppp pmssspshst laysgpvlsq 361 fnghffpgls stgilypreg tev';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count38 = 0; count38 < 1; count38++) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mafysccwvl laltwhtsay gpdqraqkkg diilgglfpi hfgvaakdqd lksrpesvec 61 irynfrgfrw lqamifaiee insspallpn ltlgyrifdt cntvskalea tlsfvaqnki 121 dslnldefcn csehipstia vvgatgsgvs tavanllglf yipqvsyass srllsnknqf 181 ksflrtipnd ehqatamadi ieyfrwnwvg tiaadddygr pgiekfreea eerdicidfs 241 elisqysdee eiqhvveviq nstakvivvf ssgpdlepli keivrrnitg kiwlaseawa 301 sssliampqy fhvvggtigf alkagqipgf reflkkvhpr ksvhngfake fweetfnchl 361 qegakgplpv dtflrghees gdrfsnssta frplctgden issvetpyid ythlrisynv 421 ylavysiaha lqdiytclpg rglftngsca dikkveawqv lkhlrhlnft nnmgeqvtfd 481 ecgdlvgnys iinwhlsped gsivfkevgy ynvyakkger lfineekilw sgfsrepltf 541 vlsvlqvpfs ncsrdclagt rkgiiegept ccfecvecpd geysdetdas acnkcpddfw 601 snenhtscia keieflswte pfgialtlfa vlgifltafv lgvfikfrnt pivkatnrel 661 sylllfsllc cfssslffig epqdwtcrlr qpafgisfvl ciscilvktn rvllvfeaki 721 ptsfhrkwwg lnlqfllvfl ctfmqivicv iwlytappss yrnqeledei ifitchegsl 781 malgfligyt cllaaicfff afksrklpen fneakfitfs mliffivwis fipayastyg 841 kfvsavevia ilaasfglla ciffnkiyii lfkpsrntie evrcstaaha fkvaaratlr 901 rsnvsrkrss slggstgstp sssissksns edpfpqperq kqqqplaltq qeqqqqpltl 961 pqqqrsqqqp rckqkvifgs gtvtfslsfd epqknamahr nsthqnslea qkssdtltrh 1021 epllplqcge tdldltvqet glqgpvggdq rpevedpeel spalvvsssq sfvisgggst 1081 vtenvvns';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count40 = 0; count40 < 1; count40++) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtgatvsvcl lfimsvcsac yisncpiggk rsvmdaplrk cmscgpgdrg hcfgpsiccg 61 dgfgcllgsp esaacveeny lltpcqaggr pcgsdgghca tsglccdaes cttdqscfme 121 engddqsspl evsdpadill rllhlaghts hrihq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count42 = 0; count42 < 1; count42++) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtgatvsvcl lfimsvcsac yisncpiggk rsvmdaplrk cmscgpgdrg hcfgpsiccg 61 dgfgcllgsp esaacveeny lltpcqaggr pcgsdgghca tsglccdaes cttdqscfme 121 engddqsspl evsdpadill rllhlaghts hrihq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count44 = 0; count44 < 1; count44++) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtgatvsvcl lfimsvcsac yisncpiggk rsvmdaplrk cmscgpgdrg hcfgpsiccg 61 dgfgcllgsp esaacveeny lltpcqaggr pcgsdgghca tsglccdaes cttdqscfme 121 engddqsspl evsdpadill rllhlaghts hrihq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count46 = 0; count46 < 1; count46++) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtgatvsvcl lfimsvcsac yisncpiggk rsvmdaplrk cmscgpgdrg hcfgpsiccg 61 dgfgcllgsp esaacveeny lltpcqaggr pcgsdgghca tsglccdaes cttdqscfme 121 engddqsspl evsdpadill rllhlaghts hrihq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['ionocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function lymphangiocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count48 = 0; count48 < 1; count48++) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 marcfslvll ltsiwttrll vqgslraeel siqvscrimg itlvskkanq qlnfteakea 61 crllglslag kdqvetalka sfetcsygwv gdgfvvisri spnpkcgkng vgvliwkvpv 121 srqfaaycyn ssdtwtnsci peiittkdpi fntqtatqtt efivsdstys vaspystipa 181 ptttppapas tsiprrkkli cvtevfmets tmstetepfv enkaafknea agfggvptal 241 lvlallffga aaglgfcyvk ryvkafpftn knqqkemiet kvvkeekand snpneeskkt 301 dknpeesksp skttvrclea ev';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count50 = 0; count50 < 1; count50++) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mwkvsallfv lgsaslwvla egastgqped dtettglegg vampgaeddv vtpgtsedry 61 ksglttlvat svnsvtgiri edlptsestv haqeqspsat asnvatshst ekvdgdtqtt 121 vekdglstvt lvgiivgvll aigfigaiiv vvmrkmsgry sp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count52 = 0; count52 < 1; count52++) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtdrqtdtap spsyhllpgr rrtvdaaasr gqgpepapgg gvegvgargv alklfvqllg 61 csrfggavvr ageaepsgaa rsassgreep qpeegeeeee keeergpqwr lgarkpgswt 121 geaavcadsa paarapqala rasgrggrva rrgaeesgpp hspsrrgsas ragpgraset 181 mnfllswvhw slalllylhh akwsqaapma egggqnhhev vkfmdvyqrs ychpietlvd 241 ifqeypdeie yifkpscvpl mrcggccnde glecvptees nitmqimrik phqgqhigem 301 sflqhnkcec rpkkdrarqe kksvrgkgkg qkrkrkksry kswsvyvgar cclmpwslpg 361 phpcgpcser rkhlfvqdpq tckcsckntd srckarqlel nertcrcdkp rr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count54 = 0; count54 < 1; count54++) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mhllgffsva csllaaallp gpreapaaaa afesgldlsd aepdageata yaskdleeql 61 rsvssvdelm tvlypeywkm ykcqlrkggw qhnreqanln srteetikfa aahynteilk 121 sidnewrktq cmprevcidv gkefgvatnt ffkppcvsvy rcggccnseg lqcmntstsy 181 lsktlfeitv plsqgpkpvt isfanhtscr cmskldvyrq vhsiirrslp atlpqcqaan 241 ktcptnymwn nhicrclaqe dfmfssdagd dstdgfhdic gpnkeldeet cqcvcraglr 301 pascgphkel drnscqcvck nklfpsqcga nrefdentcq cvckrtcprn qplnpgkcac 361 ectespqkcl lkgkkfhhqt cscyrrpctn rqkacepgfs yseevcrcvp sywkrpqms';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count56 = 0; count56 < 1; count56++) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlltliillp vvskfsfvsl sapqhwscpe gtlagngnst cvgpapflif shgnsifrid 61 tegtnyeqlv vdagvsvimd fhynekriyw vdlerqllqr vflngsrqer vcnieknvsg 121 mainwineev iwsnqqegii tvtdmkgnns hillsalkyp anvavdpver fifwssevag 181 slyradldgv gvkalletse kitavsldvl dkrlfwiqyn regsnslics cdydggsvhi 241 skhptqhnlf amslfgdrif ystwkmktiw iankhtgkdm vrinlhssfv plgelkvvhp 301 laqpkaeddt wepeqklckl rkgncsstvc gqdlqshlcm caegyalsrd rkycedvnec 361 afwnhgctlg ckntpgsyyc tcpvgfvllp dgkrchqlvs cprnvsecsh dcvltsegpl 421 cfcpegsvle rdgktcsgcs spdnggcsql cvplspvswe cdcfpgydlq ldekscaasg 481 pqpfllfans qdirhmhfdg tdygtllsqq mgmvyaldhd pvenkiyfah talkwieran 541 mdgsqrerli eegvdvpegl avdwigrrfy wtdrgkslig rsdlngkrsk iitkenisqp 601 rgiavhpmak rlfwtdtgin priessslqg lgrlviassd liwpsgitid fltdklywcd 661 akqsvieman ldgskrrrlt qndvghpfav avfedyvwfs dwampsvmrv nkrtgkdrvr 721 lqgsmlkpss lvvvhplakp gadpclyqng gcehickkrl gtawcscreg fmkasdgktc 781 laldghqlla ggevdlknqv tpldilsktr vsednitesq hmlvaeimvs dqddcapvgc 841 smyarciseg edatcqclkg fagdgklcsd idecemgvpv cppasskcin teggyvcrcs 901 egyqgdgihc ldidecqlge hscgenasct nteggytcmc agrlsepgli cpdstppphl 961 reddhhysvr nsdsecplsh dgyclhdgvc myiealdkya cncvvgyige rcqyrdlkww 1021 elrhaghgqq qkvivvavcv vvlvmlllls lwgahyyrtq kllsknpknp yeessrdvrs 1081 rrpadtedgm sscpqpwfvv ikehqdlkng gqpvagedgq aadgsmqpts wrqepqlcgm 1141 gteqgcwipv ssdkgscpqv mersfhmpsy gtqtleggve kphsllsanp lwqqraldpp 1201 hqmeltq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['lymphangiocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function mucocyte() {

orocyte();

}

function orocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count58 = 0; count58 < 1; count58++) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdkfwwhaaw glclvplsla qidlnitcrf agvfhvekng rysisrteaa dlckafnstl 61 ptmaqmekal sigfetcryg fieghvvipr ihpnsicaan ntgvyiltsn tsqydtycfn 121 asappeedct svtdlpnafd gpititivnr dgtryvqkge yrtnpediyp snptdddvss 181 gssserssts ggyifytfst vhpipdedsp witdstdrip attlmstsat atetatkrqe 241 twdwfswlfl psesknhlht ttqmagtssn tisagwepne enederdrhl sfsgsgiddd 301 edfisstist tprafdhtkq nqdwtqwnps hsnpevllqt ttrmtdvdrn gttayegnwn 361 peahpplihh ehheeeetph ststiqatps stteetatqk eqwfgnrwhe gyrqtpkeds 421 hsttgtaaas ahtshpmqgr ttpspedssw tdffnpishp mgrghqagrr mdmdsshsit 481 lqptanpntg lvedldrtgp lsmttqqsns qsfstshegl eedkdhptts tltssnrndv 541 tggrrdpnhs egsttllegy tshyphtkes rtfipvtsak tgsfgvtavt vgdsnsnvnr 601 slsgdqdtfh psggshtthg sesdghshgs qegganttsg pirtpqipew liilasllal 661 alilavciav nsrrrcgqkk klvinsgnga vedrkpsgln geasksqemv hlvnkesset 721 pdqfmtadet rnlqnvdmki gv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count60 = 0; count60 < 1; count60++) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count62 = 0; count62 < 1; count62++) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mdffrvvenq qppatmplnv sftnrnydld ydsvqpyfyc deeenfyqqq qqselqppap 61 sediwkkfel lptpplspsr rsglcspsyv avtpfslrgd ndggggsfst adqlemvtel 121 lggdmvnqsf icdpddetfi kniiiqdcmw sgfsaaaklv seklasyqaa rkdsgspnpa 181 rghsvcstss lylqdlsaaa secidpsvvf pyplndsssp kscasqdssa fspssdslls 241 stesspqgsp eplvlheetp pttssdseee qedeeeidvv svekrqapgk rsesgspsag 301 ghskpphspl vlkrchvsth qhnyaappst rkdypaakrv kldsvrvlrq isnnrkctsp 361 rssdteenvk rrthnvlerq rrnelkrsff alrdqipele nnekapkvvi lkkatayils 421 vqaeeqklis eedllrkrre qlkhkleqlr nsca';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count64 = 0; count64 < 1; count64++) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count66 = 0; count66 < 1; count66++) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maghlasdfa fspppggggd gpggpepgwv dprtwlsfqg ppggpgigpg vgpgsevwgi 61 ppcpppyefc ggmaycgpqv gvglvpqggl etsqpegeag vgvesnsdga spepctvtpg 121 avklekekle qnpeesqdik alqkeleqfa kllkqkritl gytqadvglt lgvlfgkvfs 181 qtticrfeal qlsfknmckl rpllqkwvee adnnenlqei ckaetlvqar krkrtsienr 241 vrgnlenlfl qcpkptlqqi shiaqqlgle kdvvrvwfcn rrqkgkrsss dyaqredfea 301 agspfsggpv sfplapgphf gtpgygsphf talyssvpfp egeafppvsv ttlgspmhsn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['orocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function procyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count68 = 0; count68 < 1; count68++) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mevqlglgrv yprppsktyr gafqnlfqsv reviqnpgpr hpeaasaapp gasllllqqq 61 qqqqqqqqqq qqqqqqqqqq etsprqqqqq qgedgspqah rrgptgylvl deeqqpsqpq 121 salechperg cvpepgaava askglpqqlp appdeddsaa pstlsllgpt fpglsscsad 181 lkdilseast mqllqqqqqe avsegsssgr areasgapts skdnylggts tisdnakelc 241 kavsvsmglg vealehlspg eqlrgdcmya pllgvppavr ptpcaplaec kgsllddsag 301 kstedtaeys pfkggytkgl egeslgcsgs aaagssgtle lpstlslyks galdeaaayq 361 srdyynfpla lagppppppp phpharikle npldygsawa aaaaqcrygd laslhgagaa 421 gpgsgspsaa assswhtlft aeegqlygpc gggggggggg gggggggggg gggeagavap 481 ygytrppqgl agqesdftap dvwypggmvs rvpypsptcv ksemgpwmds ysgpygdmrl 541 etardhvlpi dyyfppqktc licgdeasgc hygaltcgsc kvffkraaeg kqkylcasrn 601 dctidkfrrk ncpscrlrkc yeagmtlgar klkklgnlkl qeegeasstt spteettqkl 661 tvshiegyec qpiflnvlea iepgvvcagh dnnqpdsfaa llsslnelge rqlvhvvkwa 721 kalpgfrnlh vddqmaviqy swmglmvfam gwrsftnvns rmlyfapdlv fneyrmhksr 781 mysqcvrmrh lsqefgwlqi tpqeflcmka lllfsiipvd glknqkffde lrmnyikeld 841 riiackrknp tscsrrfyql tklldsvqpi arelhqftfd llikshmvsv dfpemmaeii 901 svqvpkilsg kvkpiyfhtq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count70 = 0; count70 < 1; count70++) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlgtvkmegh etsdwnsyya dtqeayssvp vsnmnsglgs mnsmntymtm ntmttsgnmt 61 pasfnmsyan pglgaglspg avagmpggsa gamnsmtaag vtamgtalsp sgmgamgaqq 121 aasmnglgpy aaamnpcmsp mayapsnlgr sraggggdak tfkrsyphak ppysyislit 181 maiqqapskm ltlseiyqwi mdlfpyyrqn qqrwqnsirh slsfndcfvk varspdkpgk 241 gsywtlhpds gnmfengcyl rrqkrfkcek qpgagggggs gsggsgakgg pesrkdpsga 301 snpsadsplh rgvhgktgql egapapgpaa spqtldhsga tatggaselk tpasstappi 361 ssgpgalasv pashpahgla phesqlhlkg dphysfnhpf sinnlmssse qqhkldfkay 421 eqalqyspyg stlpaslplg sasvttrspi epsalepayy qgvysrpvln ts';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count72 = 0; count72 < 1; count72++) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlrvpeprpg eakaegaapp tpskpltsfl iqdilrdgaq rqggrtssqr qrdpepepep 61 epeggrsrag aqndqlstgp raapeeaetl aeteperhlg sylldsents galprlpqtp 121 kqpqkrsraa fshtqviele rkfshqkyls aperahlakn lkltetqvki wfqnrryktk 181 rkqlsselgd lekhsslpal keeafsrasl vsvynsypyy pylycvgsws pafw';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['procyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function salivacyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count74 = 0; count74 < 1; count74++) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnlldpfmkm tdeqekglsg apsptmseds agspcpsgsg sdtentrpqe ntfpkgepdl 61 kkeseedkfp vcireavsqv lkgydwtlvp mpvrvngssk nkphvkrpmn afmvwaqaar 121 rkladqyphl hnaelsktlg klwrllnese krpfveeaer lrvqhkkdhp dykyqprrrk 181 svkngqaeae eateqthisp naifkalqad sphsssgmse vhspgehsgq sqgpptpptt 241 pktdvqpgka dlkregrplp eggrqppidf rdvdigelss dvisnietfd vnefdqylpp 301 nghpgvpath gqvtytgsyg isstaatpas aghvwmskqq apppppqqpp qappapqapp 361 qpqaappqqp aappqqpqah tlttlssepg qsqrthikte qlspshyseq qqhspqqiay 421 spfnlphysp syppitrsqy dytdhqnsss yyshaagqgt glystftymn paqrpmytpi 481 adtsgvpsip qthspqhweq pvytqltrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count76 = 0; count76 < 1; count76++) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnlldpfmkm tdeqekglsg apsptmseds agspcpsgsg sdtentrpqe ntfpkgepdl 61 kkeseedkfp vcireavsqv lkgydwtlvp mpvrvngssk nkphvkrpmn afmvwaqaar 121 rkladqyphl hnaelsktlg klwrllnese krpfveeaer lrvqhkkdhp dykyqprrrk 181 svkngqaeae eateqthisp naifkalqad sphsssgmse vhspgehsgq sqgpptpptt 241 pktdvqpgka dlkregrplp eggrqppidf rdvdigelss dvisnietfd vnefdqylpp 301 nghpgvpath gqvtytgsyg isstaatpas aghvwmskqq apppppqqpp qappapqapp 361 qpqaappqqp aappqqpqah tlttlssepg qsqrthikte qlspshyseq qqhspqqiay 421 spfnlphysp syppitrsqy dytdhqnsss yyshaagqgt glystftymn paqrpmytpi 481 adtsgvpsip qthspqhweq pvytqltrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count78 = 0; count78 < 1; count78++) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqarysvssp nslgvvpylg geqsyyraaa aaagggytam papmsvyshp ahaeqypggm 61 araygpytpq pqpkdmvkpp ysyialitma iqnapdkkit lngiyqfimd rfpfyrdnkq 121 gwqnsirhnl slnecfvkvp rddkkpgkgs ywtldpdsyn mfengsflrr rrrfkkkdav 181 kdkeekdrlh lkeppppgrq pppappeqad gnapgpqppp vriqdikten gtcpsppqpl 241 spaaalgsgs aaavpkiesp dssssslssg ssppgslpsa rplsldgads appppapsap 301 pphhsqgfsv dnimtslrgs pqsaaaelss gllasaaass ragiapplal gayspgqssl 361 ysspcsqtss agssgggggg agaaggagga gtyhcnlqam slyaagergg hlqgapggag 421 gsavddplpd yslppvtsss ssslshgggg ggggggqeag hhpaahqgrl tswylnqagg 481 dlghlasaaa aaaaagypgq qqnfhsvrem fesqriglnn spvngnsscq mafpssqsly 541 rtsgafvydc skf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count80 = 0; count80 < 1; count80++) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqarysvssp nslgvvpylg geqsyyraaa aaagggytam papmsvyshp ahaeqypggm 61 araygpytpq pqpkdmvkpp ysyialitma iqnapdkkit lngiyqfimd rfpfyrdnkq 121 gwqnsirhnl slnecfvkvp rddkkpgkgs ywtldpdsyn mfengsflrr rrrfkkkdav 181 kdkeekdrlh lkeppppgrq pppappeqad gnapgpqppp vriqdikten gtcpsppqpl 241 spaaalgsgs aaavpkiesp dssssslssg ssppgslpsa rplsldgads appppapsap 301 pphhsqgfsv dnimtslrgs pqsaaaelss gllasaaass ragiapplal gayspgqssl 361 ysspcsqtss agssgggggg agaaggagga gtyhcnlqam slyaagergg hlqgapggag 421 gsavddplpd yslppvtsss ssslshgggg ggggggqeag hhpaahqgrl tswylnqagg 481 dlghlasaaa aaaaagypgq qqnfhsvrem fesqriglnn spvngnsscq mafpssqsly 541 rtsgafvydc skf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['salivacyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function sudoricyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count82 = 0; count82 < 1; count82++) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count84 = 0; count84 < 1; count84++) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count86 = 0; count86 < 1; count86++) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.940410259e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count88 = 0; count88 < 1; count88++) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.002289533e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count90 = 0; count90 < 1; count90++) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.064168806e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count92 = 0; count92 < 1; count92++) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 6.12604808e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count94 = 0; count94 < 1; count94++) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mrdrlpdlta crknddgdtv vvvekdhfmd dffhqveeir nsidkitqyv eevkknhsii 61 lsapnpegki keeledlnke ikktankiaa klkaieqsfd qdesgnrtsv dlrirrtqhs 121 vlsrkfveam aeyneaqtlf rerskgriqr qleitgrttt ddeleemles gkpsiftsdi 181 isdsqitrqa lneiesrhkd imkletsire lhemfmdmam fvetqgemin niernvmnat 241 dyvehakeet kkaikyqska rrkkwiiiav svvlvvyrlf glsleyvvrs aaslpgwgn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['sudoricyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function thymocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count96 = 0; count96 < 1; count96++) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvslpppqsd vtlpgptrle gerqgdlmqa pglpgspapq skhagfscss fvsdgppert 61 pslpphspri aspgpeqvqg hcpagpgpgp frlspsdkyp gfgfeeaaas spgrflkgsh 121 apfhpykrpf hedvfpeaet tlalkghsfk tpgpleafee ipvdvaeaea flpgfsaeaw 181 cnglpypsqe hgpqvlgsev kvkppvlesg agmfcyqppl qhmycssqpp fhqyspgggs 241 ypipylgssh yqyqrmapqa stdghqplfp kpiysysili fmalknsktg slpvseiynf 301 mtehfpyfkt apdgwknsvr hnlslnkcfe kvenksgsss rkgclwalnp akidkmqeel 361 qkwkrkdpia vrksmakpee ldsligdkre klgspllgcp ppglsgsgpi rplappagls 421 pplhslhpap gpipgknplq dllmghtpsc ygqtylhlsp glappgppqp lfpqpdghle 481 lraqpgtpqd splpahtpps hsakllaeps partmhdtll pdgdlgtdld ainpsltdfd 541 fqgnlweqlk ddslaldplv lvtssptsss mpppqppphc fppgpcltet gsgagdlaap 601 gsggsgalgd lhlttlysaf meleptppta pagpsvylsp sskpvala';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count98 = 0; count98 < 1; count98++) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvslpppqsd vtlpgptrle gerqgdlmqa pglpgspapq skhagfscss fvsdgppert 61 pslpphspri aspgpeqvqg hcpagpgpgp frlspsdkyp gfgfeeaaas spgrflkgsh 121 apfhpykrpf hedvfpeaet tlalkghsfk tpgpleafee ipvdvaeaea flpgfsaeaw 181 cnglpypsqe hgpqvlgsev kvkppvlesg agmfcyqppl qhmycssqpp fhqyspgggs 241 ypipylgssh yqyqrmapqa stdghqplfp kpiysysili fmalknsktg slpvseiynf 301 mtehfpyfkt apdgwknsvr hnlslnkcfe kvenksgsss rkgclwalnp akidkmqeel 361 qkwkrkdpia vrksmakpee ldsligdkre klgspllgcp ppglsgsgpi rplappagls 421 pplhslhpap gpipgknplq dllmghtpsc ygqtylhlsp glappgppqp lfpqpdghle 481 lraqpgtpqd splpahtpps hsakllaeps partmhdtll pdgdlgtdld ainpsltdfd 541 fqgnlweqlk ddslaldplv lvtssptsss mpppqppphc fppgpcltet gsgagdlaap 601 gsggsgalgd lhlttlysaf meleptppta pagpsvylsp sskpvala';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count100 = 0; count100 < 1; count100++) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvslpppqsd vtlpgptrle gerqgdlmqa pglpgspapq skhagfscss fvsdgppert 61 pslpphspri aspgpeqvqg hcpagpgpgp frlspsdkyp gfgfeeaaas spgrflkgsh 121 apfhpykrpf hedvfpeaet tlalkghsfk tpgpleafee ipvdvaeaea flpgfsaeaw 181 cnglpypsqe hgpqvlgsev kvkppvlesg agmfcyqppl qhmycssqpp fhqyspgggs 241 ypipylgssh yqyqrmapqa stdghqplfp kpiysysili fmalknsktg slpvseiynf 301 mtehfpyfkt apdgwknsvr hnlslnkcfe kvenksgsss rkgclwalnp akidkmqeel 361 qkwkrkdpia vrksmakpee ldsligdkre klgspllgcp ppglsgsgpi rplappagls 421 pplhslhpap gpipgknplq dllmghtpsc ygqtylhlsp glappgppqp lfpqpdghle 481 lraqpgtpqd splpahtpps hsakllaeps partmhdtll pdgdlgtdld ainpsltdfd 541 fqgnlweqlk ddslaldplv lvtssptsss mpppqppphc fppgpcltet gsgagdlaap 601 gsggsgalgd lhlttlysaf meleptppta pagpsvylsp sskpvala';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count102 = 0; count102 < 1; count102++) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvslpppqsd vtlpgptrle gerqgdlmqa pglpgspapq skhagfscss fvsdgppert 61 pslpphspri aspgpeqvqg hcpagpgpgp frlspsdkyp gfgfeeaaas spgrflkgsh 121 apfhpykrpf hedvfpeaet tlalkghsfk tpgpleafee ipvdvaeaea flpgfsaeaw 181 cnglpypsqe hgpqvlgsev kvkppvlesg agmfcyqppl qhmycssqpp fhqyspgggs 241 ypipylgssh yqyqrmapqa stdghqplfp kpiysysili fmalknsktg slpvseiynf 301 mtehfpyfkt apdgwknsvr hnlslnkcfe kvenksgsss rkgclwalnp akidkmqeel 361 qkwkrkdpia vrksmakpee ldsligdkre klgspllgcp ppglsgsgpi rplappagls 421 pplhslhpap gpipgknplq dllmghtpsc ygqtylhlsp glappgppqp lfpqpdghle 481 lraqpgtpqd splpahtpps hsakllaeps partmhdtll pdgdlgtdld ainpsltdfd 541 fqgnlweqlk ddslaldplv lvtssptsss mpppqppphc fppgpcltet gsgagdlaap 601 gsggsgalgd lhlttlysaf meleptppta pagpsvylsp sskpvala';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['thymocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function vesicocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count104 = 0; count104 < 1; count104++) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtasvllhpr wieptvmfly dnggglvade lnknmegaaa aaaaaaaaaa agaggggfph 61 paaaaaggnf svaaaaaaaa aaaanqcrnl mahpaplapg aasayssapg eappsaaaaa 121 aaaaaaaaaa aaasssggpg pagpagaeaa kqcspcsaaa qsssgpaalp ygyfgsgyyp 181 carmgphpna ikscaqpasa aaaaafadky mdtagpaaee fssrakefaf yhqgyaagpy 241 hhhqpmpgyl dmpvvpglgg pgesrheplg lpmesyqpwa lpngwngqmy cpkeqaqpph 301 lwkstlpdvv shpsdassyr rgrkkrvpyt kvqlkelere yatnkfitkd krrrisattn 361 lserqvtiwf qnrrvkekkv inklktts';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count106 = 0; count106 < 1; count106++) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msragswdmd glradgggag gapassssss vaaaaasgqc rgflsapvfa gthsgraaaa 61 aaaaaaaaaa asgfaypgts ertgssssss ssavvaarpe appakecpap tpaaaaaapp 121 sapalgygyh fgngyyscrm shgvglqqna lkssphaslg gfpvekymdv sglasssvpa 181 nevparakev sfyqgytspy qhvpgyidmv stfgsgeprh eayismegyq swtlangwns 241 qvyctkdqpq gshfwkssfp gdvalnqpdm cvyrrgrkkr vpytklqlke leneyainkf 301 inkdkrrris aatnlserqv tiwfqnrrvk dkkivsklkd tvs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count108 = 0; count108 < 1; count108++) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mktknrpprr rapvqdteat pgegtpdgsl pnpgpepakg lrsrparaaa rapgegrrrr 61 pgpsgpggrr dssiqrrles nererqrmhk lnnafqalre viphvradkk lskietltla 121 knyiksltat iltmsssrlp glegpgpkly qhyqqqqqva ggalgateaq pqghlqryst 181 qihsfregt';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['vesicocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function default\_cell\_body() {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count2 = 0; count2 < generation\_parameter\_x; count2++) {

for (var count = 0; count < generation\_parameter\_z; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count4 = 0; count4 < generation\_parameter\_x; count4++) {

for (var count3 = 0; count3 < generation\_parameter\_z; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count6 = 0; count6 < generation\_parameter\_x; count6++) {

for (var count5 = 0; count5 < generation\_parameter\_z; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count8 = 0; count8 < generation\_parameter\_x; count8++) {

for (var count7 = 0; count7 < generation\_parameter\_z; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count10 = 0; count10 < generation\_parameter\_x; count10++) {

for (var count9 = 0; count9 < generation\_parameter\_3; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count12 = 0; count12 < generation\_parameter\_x; count12++) {

for (var count11 = 0; count11 < generation\_parameter\_3; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 - 0);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count14 = 0; count14 < generation\_parameter\_x; count14++) {

for (var count13 = 0; count13 < generation\_parameter\_3; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count16 = 0; count16 < generation\_parameter\_x; count16++) {

for (var count15 = 0; count15 < generation\_parameter\_3; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count18 = 0; count18 < generation\_parameter\_z; count18++) {

for (var count17 = 0; count17 < generation\_parameter\_3; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count20 = 0; count20 < generation\_parameter\_z; count20++) {

for (var count19 = 0; count19 < generation\_parameter\_3; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count22 = 0; count22 < generation\_parameter\_z; count22++) {

for (var count21 = 0; count21 < generation\_parameter\_3; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count24 = 0; count24 < generation\_parameter\_z; count24++) {

for (var count23 = 0; count23 < generation\_parameter\_3; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 3.15584295e+28 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

nucleoid\_size\_z = 5.507255344e+29;

nucleoid\_size\_y = 4.331549147e+29;

nucleoid();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.25973825e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count25 = 0; count25 < 8; count25++) {

mitochondrion();

cursor1x = cursor1x - 6.187927353e+28;

}

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

vital\_components();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 1.299464744e+29 \* 1;

vital\_components2();

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 4.709012716e+29 \* 1;

cursor1z = spawnerz + 2.475170941e+29 \* 1;

vital\_components3();

}

function omnicyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

default\_cell\_fluids();

cells\_list.push(['omnicyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function erythrocyte() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 4.950341881e+29, spawnery - 2.47517094e+29, spawnerx - 4.950341881e+29];

make = true;

screen\_cells();

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 1.856378206e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(4.331549147e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count = 0; count < generation\_parameter\_x; count++) {

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(4.331549147e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count2 = 0; count2 < generation\_parameter\_x; count2++) {

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (1.856378206e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(4.331549147e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count3 = 0; count3 < generation\_parameter\_x; count3++) {

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (6.187927353e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(4.331549147e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count4 = 0; count4 < generation\_parameter\_x; count4++) {

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (6.187927353e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count6 = 0; count6 < generation\_parameter\_x; count6++) {

for (var count5 = 0; count5 < generation\_parameter\_3; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (6.187927353e+28 \* 1 - 0);

cursor1z = spawnerz + 4.331549147e+29 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count8 = 0; count8 < generation\_parameter\_x; count8++) {

for (var count7 = 0; count7 < generation\_parameter\_3; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count10 = 0; count10 < generation\_parameter\_x; count10++) {

for (var count9 = 0; count9 < generation\_parameter\_3; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + (4.331549147e+29 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

for (var count12 = 0; count12 < generation\_parameter\_x; count12++) {

for (var count11 = 0; count11 < generation\_parameter\_3; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

for (var count13 = 0; count13 < generation\_parameter\_3; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (4.331549147e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

for (var count14 = 0; count14 < generation\_parameter\_3; count14++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

for (var count15 = 0; count15 < generation\_parameter\_3; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 6.187927353e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(4.331549147e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(4.331549147e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (4.640945515e+29 - 0)) {

for (var count16 = 0; count16 < generation\_parameter\_3; count16++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 3.279601497e+27;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 6.187927353e+28 \* 1;

generation\_parameter\_x = Math.round(5384.61538462);

generation\_parameter\_3 = Math.round(1818.18181818);

generation\_parameter\_z = Math.round(132.075471698);

for (var count19 = 0; count19 < generation\_parameter\_3; count19++) {

for (var count18 = 0; count18 < generation\_parameter\_x; count18++) {

for (var count17 = 0; count17 < generation\_parameter\_z; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = ' 1 mvhltpeeks avtalwgkvn vdevggealg rllvvypwtq rffesfgdls tpdavmgnpk 61 vkahgkkvlg afsdglahld nlkgtfatls elhcdklhvd penfrllgnv lvcvlahhfg 121 keftppvqaa yqkvvagvan alahkyh';

protein();

} else {

peptidesequence = '1 mvlspadktn vkaawgkvga hageygaeal ermflsfptt ktyfphfdls hgsaqvkghg 61 kkvadaltna vahvddmpna lsalsdlhah klrvdpvnfk llshcllvtl aahlpaeftp 121 avhasldkfl asvstvltsk yr';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 6.806720089e+25;

cursor1x = spawnerx - 3.974505739e+29 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 6.187927353e+28 \* 1;

generation\_parameter\_x = Math.round(909.090909091);

generation\_parameter\_3 = Math.round(1818.18181818);

generation\_parameter\_z = Math.round(200);

for (var count22 = 0; count22 < generation\_parameter\_3; count22++) {

for (var count21 = 0; count21 < generation\_parameter\_x; count21++) {

for (var count20 = 0; count20 < generation\_parameter\_z; count20++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

heme();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((5 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count25 = 0; count25 < generation\_parameter\_3; count25++) {

for (var count24 = 0; count24 < generation\_parameter\_x; count24++) {

for (var count23 = 0; count23 < generation\_parameter\_z; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

glucose();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((10 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count28 = 0; count28 < generation\_parameter\_3; count28++) {

for (var count27 = 0; count27 < generation\_parameter\_x; count27++) {

for (var count26 = 0; count26 < generation\_parameter\_z; count26++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_fat();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((4.9 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count31 = 0; count31 < generation\_parameter\_3; count31++) {

for (var count30 = 0; count30 < generation\_parameter\_x; count30++) {

for (var count29 = 0; count29 < generation\_parameter\_z; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_amino\_acid();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((4.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count34 = 0; count34 < generation\_parameter\_3; count34++) {

for (var count33 = 0; count33 < generation\_parameter\_x; count33++) {

for (var count32 = 0; count32 < generation\_parameter\_z; count32++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_ribonucleotide();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((9.95 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count37 = 0; count37 < generation\_parameter\_3; count37++) {

for (var count36 = 0; count36 < generation\_parameter\_x; count36++) {

for (var count35 = 0; count35 < generation\_parameter\_z; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

ADP();

} else if (randomizer <= 20) {

AMP();

} else if (randomizer <= 30) {

FAD();

} else if (randomizer <= 35) {

NAD();

} else if (randomizer <= 40) {

NADH();

} else if (randomizer <= 50) {

FADH();

} else {

ATP();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((4.8 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count40 = 0; count40 < generation\_parameter\_3; count40++) {

for (var count39 = 0; count39 < generation\_parameter\_x; count39++) {

for (var count38 = 0; count38 < generation\_parameter\_z; count38++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_vitamin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.922068735e+25;

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 1.23758547e+26 \* 1;

cursor1y = spawnery - 1 \* 1;

cursor1z = spawnerz + 1.23758547e+26 \* 1;

generation\_parameter\_x = Math.round(4.950341882e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(2.475170941e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(4.950341882e+29 / density\_parameter\_z - 3);

for (var count43 = 0; count43 < generation\_parameter\_3; count43++) {

for (var count42 = 0; count42 < generation\_parameter\_x; count42++) {

for (var count41 = 0; count41 < generation\_parameter\_z; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

if (cursor1x < spawnerx - 4.640945515e+29 \* 1) {

randomize\_ECF();

} else if (cursor1y < spawnery - 1.856378206e+29 \* 1) {

randomize\_ECF();

} else if (cursor1z > spawnerz + 4.640945515e+29 \* 1) {

randomize\_ECF();

} else if (cursor1x > spawnerx - 3.093963676e+28 \* 1) {

randomize\_ECF();

} else if (cursor1y > spawnery - 6.187927353e+28 \* 1) {

randomize\_ECF();

} else if (cursor1z < spawnerz + 3.093963676e+28 \* 1) {

randomize\_ECF();

} else {

randomize\_cytosol();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

list\_cells.push(['erythrocyte', spawnerz, spawnery, spawnerx, spawnerz + 4.950341881e+29, spawnery - 2.47517094e+29, spawnerx - 4.950341881e+29]);

}

}

function melanocyte() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_cells();

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count44 = 0; count44 < generation\_parameter\_x; count44++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count45 = 0; count45 < generation\_parameter\_x; count45++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count46 = 0; count46 < generation\_parameter\_x; count46++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count47 = 0; count47 < generation\_parameter\_x; count47++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count49 = 0; count49 < generation\_parameter\_x; count49++) {

for (var count48 = 0; count48 < generation\_parameter\_3; count48++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 - 0);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count51 = 0; count51 < generation\_parameter\_x; count51++) {

for (var count50 = 0; count50 < generation\_parameter\_3; count50++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count53 = 0; count53 < generation\_parameter\_x; count53++) {

for (var count52 = 0; count52 < generation\_parameter\_3; count52++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count55 = 0; count55 < generation\_parameter\_x; count55++) {

for (var count54 = 0; count54 < generation\_parameter\_3; count54++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count56 = 0; count56 < generation\_parameter\_3; count56++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count57 = 0; count57 < generation\_parameter\_3; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count58 = 0; count58 < generation\_parameter\_3; count58++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count59 = 0; count59 < generation\_parameter\_3; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = Math.pow((50 \* (6.0221409e+23 / 1000)) / 1000000000000, 1 / 3);

density\_parameter\_y = density\_parameter\_x;

density\_parameter\_z = density\_parameter\_x;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 3);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 3);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 3);

for (var count62 = 0; count62 < generation\_parameter\_3; count62++) {

for (var count61 = 0; count61 < generation\_parameter\_x; count61++) {

for (var count60 = 0; count60 < generation\_parameter\_z; count60++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

melanin();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cell\_fluids();

list\_cells.push(['melanocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function dermatocyte() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_cells();

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count = 0; count < generation\_parameter\_x; count++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count2 = 0; count2 < generation\_parameter\_x; count2++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count3 = 0; count3 < generation\_parameter\_x; count3++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count4 = 0; count4 < generation\_parameter\_x; count4++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count6 = 0; count6 < generation\_parameter\_x; count6++) {

for (var count5 = 0; count5 < generation\_parameter\_3; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 - 0);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count8 = 0; count8 < generation\_parameter\_x; count8++) {

for (var count7 = 0; count7 < generation\_parameter\_3; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count10 = 0; count10 < generation\_parameter\_x; count10++) {

for (var count9 = 0; count9 < generation\_parameter\_3; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count12 = 0; count12 < generation\_parameter\_x; count12++) {

for (var count11 = 0; count11 < generation\_parameter\_3; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count13 = 0; count13 < generation\_parameter\_3; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count14 = 0; count14 < generation\_parameter\_3; count14++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count15 = 0; count15 < generation\_parameter\_3; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count16 = 0; count16 < generation\_parameter\_3; count16++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 3.279601497e+27;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 6.187927353e+28 \* 1;

generation\_parameter\_x = Math.round(5384.61538462);

generation\_parameter\_3 = Math.round(1818.18181818);

generation\_parameter\_z = Math.round(132.075471698);

for (var count19 = 0; count19 < generation\_parameter\_3; count19++) {

for (var count18 = 0; count18 < generation\_parameter\_x; count18++) {

for (var count17 = 0; count17 < generation\_parameter\_z; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = '1 msrqssvsfr sggsrsfsta saitpsvsrt sftsvsrsgg gggggfgrvs lagacgvggy 61 gsrslynlgg skrisistsg gsfrnrfgag agggygfggg agsgfgfggg agggfglggg 121 agfgggfggp gfpvcppggi qevtvnqsll tplnlqidps iqrvrteere qiktlnnkfa 181 sfidkvrfle qqnkvldtkw tllqeqgtkt vrqnleplfe qyinnlrrql dsivgergrl 241 dselrnmqdl vedfknkyed einkrttaen efvmlkkdvd aaymnkvele akvdalmdei 301 nfmkmffdae lsqmqthvsd tsvvlsmdnn rnldldsiia evkaqyeeia nrsrteaesw 361 yqtkyeelqq tagrhgddlr ntkheisemn rmiqrlraei dnvkkqcanl qnaiadaeqr 421 gelalkdarn klaeleealq kakqdmarll reyqelmntk laldveiaty rkllegeecr 481 lsgegvgpvn isvvtssvss gygsgsgygg glggglgggl ggglaggssg syyssssggv 541 glggglsvgg sgfsassgrg lgvgfgsggg ssssvkfvst tsssrksfks';

protein();

} else {

peptidesequence = '1 mttcsrqfts sssmkgscgi gggigggssr issvlaggsc rapstygggl svsssrfssg 61 gacglgggyg ggfssssssf gsgfgggygg glgaglgggf gggfaggdgl lvgsekvtmq 121 nlndrlasyl dkvraleean adlevkirdw yqrqrpaeik dyspyfktie dlrnkiltat 181 vdnanvllqi dnarlaaddf rtkyetelnl rmsveading lrrvldeltl aradlemqie 241 slkeelaylk knheeemnal rgqvggdvnv emdaapgvdl srilnemrdq yekmaeknrk 301 daeewfftkt eelnrevatn selvqsgkse iselrrtmqn leielqsqls mkaslensle 361 etkgrycmql aqiqemigsv eeqlaqlrce meqqnqeyki lldvktrleq eiatyrrlle 421 gedahlsssq fssgsqssrd vtsssrqirt kvmdvhdgkv vstheqvlrt kn';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cell\_fluids();

list\_cells.push(['dermatocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function keratinocyte() {

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_cells();

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 5.878530986e+29 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count20 = 0; count20 < generation\_parameter\_x; count20++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / density\_parameter\_z - 0);

for (var count21 = 0; count21 < generation\_parameter\_x; count21++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_1();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count22 = 0; count22 < generation\_parameter\_x; count22++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 1.546981838e+26 \* 1;

density\_parameter\_z = 4.085906752e+25;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 1.546981838e+26 \* 1);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count23 = 0; count23 < generation\_parameter\_x; count23++) {

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_2();

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = spawnerz + 3.093963676e+28 \* 1;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - (3.093963676e+28 \* 1 - 0);

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count25 = 0; count25 < generation\_parameter\_x; count25++) {

for (var count24 = 0; count24 < generation\_parameter\_3; count24++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 - 0);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count27 = 0; count27 < generation\_parameter\_x; count27++) {

for (var count26 = 0; count26 < generation\_parameter\_3; count26++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_3();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count29 = 0; count29 < generation\_parameter\_x; count29++) {

for (var count28 = 0; count28 < generation\_parameter\_3; count28++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 4.085906752e+25 \* 1;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 1.546981838e+26;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + (5.878530986e+29 \* 1 + 1.546981838e+26 \* 1);

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

for (var count31 = 0; count31 < generation\_parameter\_x; count31++) {

for (var count30 = 0; count30 < generation\_parameter\_3; count30++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_4();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

cursor1x = cursor1x - density\_parameter\_x;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count32 = 0; count32 < generation\_parameter\_3; count32++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 - 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count33 = 0; count33 < generation\_parameter\_3; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_5();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (3.093963676e+28 \* 1 + 1.546981838e+26 \* 1);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count34 = 0; count34 < generation\_parameter\_3; count34++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 1.546981838e+26;

density\_parameter\_y = 4.085906752e+25 \* 1;

density\_parameter\_z = 4.085906752e+25 \* 1;

cursor1x = spawnerx - (5.878530986e+29 \* 1 + 0);

cursor1y = spawnery - 3.093963676e+28 \* 1;

cursor1z = spawnerz + 3.093963676e+28 \* 1;

generation\_parameter\_x = Math.round(5.569134618e+29 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(5.569134618e+29 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+29 / generation\_parameter\_z - 0);

while (cursor1z < spawnerz + (5.878530986e+29 - 0)) {

for (var count35 = 0; count35 < generation\_parameter\_3; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomize\_lipid\_6();

}

cursor1y = cursor1y - density\_parameter\_y;

}

cursor1z = cursor1z + density\_parameter\_z;

cursor1y = cursor1y + generation\_parameter\_3 \* density\_parameter\_y;

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = spawnerx;

cursor1y = spawnery;

cursor1z = spawnerz;

}

density\_parameter\_x = 6.806720089e+25;

density\_parameter\_y = 6.806720089e+25;

density\_parameter\_z = 3.279601497e+27;

cursor1x = spawnerx - 3.093963676e+28 \* 1;

cursor1y = spawnery - 1.23758547e+26 \* 1;

cursor1z = spawnerz + 6.187927353e+28 \* 1;

generation\_parameter\_x = Math.round(5384.61538462);

generation\_parameter\_3 = Math.round(3600);

generation\_parameter\_z = Math.round(132.075471698);

for (var count38 = 0; count38 < generation\_parameter\_3; count38++) {

for (var count37 = 0; count37 < generation\_parameter\_x; count37++) {

for (var count36 = 0; count36 < generation\_parameter\_z; count36++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

peptidesequence = '1 msrqssvsfr sggsrsfsta saitpsvsrt sftsvsrsgg gggggfgrvs lagacgvggy 61 gsrslynlgg skrisistsg gsfrnrfgag agggygfggg agsgfgfggg agggfglggg 121 agfgggfggp gfpvcppggi qevtvnqsll tplnlqidps iqrvrteere qiktlnnkfa 181 sfidkvrfle qqnkvldtkw tllqeqgtkt vrqnleplfe qyinnlrrql dsivgergrl 241 dselrnmqdl vedfknkyed einkrttaen efvmlkkdvd aaymnkvele akvdalmdei 301 nfmkmffdae lsqmqthvsd tsvvlsmdnn rnldldsiia evkaqyeeia nrsrteaesw 361 yqtkyeelqq tagrhgddlr ntkheisemn rmiqrlraei dnvkkqcanl qnaiadaeqr 421 gelalkdarn klaeleealq kakqdmarll reyqelmntk laldveiaty rkllegeecr 481 lsgegvgpvn isvvtssvss gygsgsgygg glggglgggl ggglaggssg syyssssggv 541 glggglsvgg sgfsassgrg lgvgfgsggg ssssvkfvst tsssrksfks';

protein();

} else {

peptidesequence = '1 mttcsrqfts sssmkgscgi gggigggssr issvlaggsc rapstygggl svsssrfssg 61 gacglgggyg ggfssssssf gsgfgggygg glgaglgggf gggfaggdgl lvgsekvtmq 121 nlndrlasyl dkvraleean adlevkirdw yqrqrpaeik dyspyfktie dlrnkiltat 181 vdnanvllqi dnarlaaddf rtkyetelnl rmsveading lrrvldeltl aradlemqie 241 slkeelaylk knheeemnal rgqvggdvnv emdaapgvdl srilnemrdq yekmaeknrk 301 daeewfftkt eelnrevatn selvqsgkse iselrrtmqn leielqsqls mkaslensle 361 etkgrycmql aqiqemigsv eeqlaqlrce meqqnqeyki lldvktrleq eiatyrrlle 421 gedahlsssq fssgsqssrd vtsssrqirt kvmdvhdgkv vstheqvlrt kn';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + generation\_parameter\_x \* density\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

default\_cell\_fluids();

list\_cells.push(['keratinocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function audioblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count2 = 0; count2 < 1; count2++) {

for (var count = 0; count < 4500; count++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mprsflvksk kahsyhqprs pgpdyslrle nvpapsrads tsnaggakae prdrlspesq 61 lteapdrasa spdscegsvc erssefedfw rppspsaspa seksmcpsld eaqpfplpfk 121 pyswsglags dlrhlvqsyr pcgalergag lglfcepape pghpaalygp kraaggagag 181 apgscsagag atagpglgly gdfgsaaagl yerptaaagl lyperghglh adkgagvkve 241 sellctrlll gggsykcikc skvfstphgl evhvrrshsg trpfacemcg ktfghavsle 301 qhkavhsqer sfdckicgks fkrsstlsth llihsdtrpy pcqycgkrfh qksdmkkhtf 361 ihtgekphkc qvcgkafsqs snlithsrkh tgfkpfgcdl cgkgfqrkvd lrrhretqhg 421 lk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count4 = 0; count4 < 1; count4++) {

for (var count3 = 0; count3 < 4500; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mprsflvksk kahsyhqprs pgpdyslrle nvpapsrads tsnaggakae prdrlspesq 61 lteapdrasa spdscegsvc erssefedfw rppspsaspa seksmcpsld eaqpfplpfk 121 pyswsglags dlrhlvqsyr pcgalergag lglfcepape pghpaalygp kraaggagag 181 apgscsagag atagpglgly gdfgsaaagl yerptaaagl lyperghglh adkgagvkve 241 sellctrlll gggsykcikc skvfstphgl evhvrrshsg trpfacemcg ktfghavsle 301 qhkavhsqer sfdckicgks fkrsstlsth llihsdtrpy pcqycgkrfh qksdmkkhtf 361 ihtgekphkc qvcgkafsqs snlithsrkh tgfkpfgcdl cgkgfqrkvd lrrhretqhg 421 lk';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count6 = 0; count6 < 1; count6++) {

for (var count5 = 0; count5 < 4500; count5++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmamnskqpf gmhpvlqepk fsslhsgsea mrrvclpapq lqgnifgsfd esllaraeal 61 aavdivshgk nhpfkpdaty htmssvpcts tsstvpishp aaltshphha vhqglegdll 121 ehisptlsvs glgapehsvm paqihphhlg amghlhqamg mshphtvaph sampaclsdv 181 esdpreleaf aerfkqrrik lgvtqadvga alanlkipgv gslsqsticr fesltlshnn 241 mialkpvlqa wleeaeaayr eknskpelfn gserkrkrts iaapekrsle ayfaiqprps 301 sekiaaiaek ldlkknvvrv wfcnqrqkqk rmkysavh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count8 = 0; count8 < 1; count8++) {

for (var count7 = 0; count7 < 4500; count7++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmamnskqpf gmhpvlqepk fsslhsgsea mrrvclpapq lqgnifgsfd esllaraeal 61 aavdivshgk nhpfkpdaty htmssvpcts tsstvpishp aaltshphha vhqglegdll 121 ehisptlsvs glgapehsvm paqihphhlg amghlhqamg mshphtvaph sampaclsdv 181 esdpreleaf aerfkqrrik lgvtqadvga alanlkipgv gslsqsticr fesltlshnn 241 mialkpvlqa wleeaeaayr eknskpelfn gserkrkrts iaapekrsle ayfaiqprps 301 sekiaaiaek ldlkknvvrv wfcnqrqkqk rmkysavh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count10 = 0; count10 < 1; count10++) {

for (var count9 = 0; count9 < 4500; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msrllhaeew aevkelgdhh rqpqphhlpq pppppqppat lqarehpvyp pelslldstd 61 prawlaptlq gictaraaqy llhspelgas eaaaprdevd grgelvrrss ggassskspg 121 pvkvreqlck lkggvvvdel gcsrqrapss kqvngvqkqr rlaanarerr rmhglnhafd 181 qlrnvipsfn ndkklskyet lqmaqiyina lsellqtpsg geqpppppas cksdhhhlrt 241 aasyeggagn ataagaqqas ggsqrptppg scrtrfsapa saggysvqld alhfstfeds 301 altammaqkn lspslpgsil qpvqeenskt sprshrsdge fsphshysds deas';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count12 = 0; count12 < 1; count12++) {

for (var count11 = 0; count11 < 4500; count11++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msrllhaeew aevkelgdhh rqpqphhlpq pppppqppat lqarehpvyp pelslldstd 61 prawlaptlq gictaraaqy llhspelgas eaaaprdevd grgelvrrss ggassskspg 121 pvkvreqlck lkggvvvdel gcsrqrapss kqvngvqkqr rlaanarerr rmhglnhafd 181 qlrnvipsfn ndkklskyet lqmaqiyina lsellqtpsg geqpppppas cksdhhhlrt 241 aasyeggagn ataagaqqas ggsqrptppg scrtrfsapa saggysvqld alhfstfeds 301 altammaqkn lspslpgsil qpvqeenskt sprshrsdge fsphshysds deas';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['audioblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function epiophalmocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count14 = 0; count14 < 1; count14++) {

for (var count13 = 0; count13 < 4500; count13++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhhqqrmaal gtdkelsdll dfsamfsppv ssgkngptsl asghftgsnv edrsssgswg 61 ngghpspsrn ygdgtpydhm tsrdlgshdn lsppfvnsri qsktergsys sygresnlqg 121 chqqsllggd mdmgnpgtls ptkpgsqyyq yssnnprrrp lhssamevqt kkvrkvppgl 181 pssvyapsas tadynrdspg ypsskpatst fpssffmqdg hhssdpwsss sgmnqpgyag 241 mlgnsshipq sssycslhph erlsypshss adinsslppm stfhrsgtnh ystssctppa 301 ngtdsimanr gsgaagssqt gdalgkalas iyspdhtnns fssnpstpvg sppslsagta 361 vwsrnggqas sspnyegplh slqsriedrl erlddaihvl rnhavgpsta mpgghgdmhg 421 iigpshngam gglgsgygtg llsanrhslm vgthredgva lrgshsllpn qvpvpqlpvq 481 satspdlnpp qdpyrgmppg lqgqsvssgs seiksddegd enlqdtksse dkkldddkkd 541 iksitrsrss nnddedltpe qkaerekerr mannarerlr vrdineafke lgrmvqlhlk 601 sdkpqtklli lhqavavils leqqvrernl npkaaclkrr eeekvssepp plslagphpg 661 mgdasnhmgq m';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count16 = 0; count16 < 1; count16++) {

for (var count15 = 0; count15 < 4500; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhhqqrmaal gtdkelsdll dfsamfsppv ssgkngptsl asghftgsnv edrsssgswg 61 ngghpspsrn ygdgtpydhm tsrdlgshdn lsppfvnsri qsktergsys sygresnlqg 121 chqqsllggd mdmgnpgtls ptkpgsqyyq yssnnprrrp lhssamevqt kkvrkvppgl 181 pssvyapsas tadynrdspg ypsskpatst fpssffmqdg hhssdpwsss sgmnqpgyag 241 mlgnsshipq sssycslhph erlsypshss adinsslppm stfhrsgtnh ystssctppa 301 ngtdsimanr gsgaagssqt gdalgkalas iyspdhtnns fssnpstpvg sppslsagta 361 vwsrnggqas sspnyegplh slqsriedrl erlddaihvl rnhavgpsta mpgghgdmhg 421 iigpshngam gglgsgygtg llsanrhslm vgthredgva lrgshsllpn qvpvpqlpvq 481 satspdlnpp qdpyrgmppg lqgqsvssgs seiksddegd enlqdtksse dkkldddkkd 541 iksitrsrss nnddedltpe qkaerekerr mannarerlr vrdineafke lgrmvqlhlk 601 sdkpqtklli lhqavavils leqqvrernl npkaaclkrr eeekvssepp plslagphpg 661 mgdasnhmgq m';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count18 = 0; count18 < 1; count18++) {

for (var count17 = 0; count17 < 4500; count17++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msaalfsldg pargapwpae papfyepgra gkpgrgaepg algepgaaap amyddesaid 61 fsayidsmaa vptlelchde lfadlfnsnh kaggagplel lpggparplg pgpaaprllk 121 repdwgdgda pgsllpaqva acaqtvvsla aagqptppts pepprssprq tpapgparek 181 sagkrgpdrg speyrqrrer nniavrksrd kakrrnqemq qklvelsaen eklhqrveql 241 trdlaglrqf fkqlpsppfl paagtadcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count20 = 0; count20 < 1; count20++) {

for (var count19 = 0; count19 < 4500; count19++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 msaalfsldg pargapwpae papfyepgra gkpgrgaepg algepgaaap amyddesaid 61 fsayidsmaa vptlelchde lfadlfnsnh kaggagplel lpggparplg pgpaaprllk 121 repdwgdgda pgsllpaqva acaqtvvsla aagqptppts pepprssprq tpapgparek 181 sagkrgpdrg speyrqrrer nniavrksrd kakrrnqemq qklvelsaen eklhqrveql 241 trdlaglrqf fkqlpsppfl paagtadcr';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count22 = 0; count22 < 1; count22++) {

for (var count21 = 0; count21 < 4500; count21++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count24 = 0; count24 < 1; count24++) {

for (var count23 = 0; count23 < 4500; count23++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mnfetsrcat lqycpdpyiq rfvetpahfs wkesyyrstm sqstqtnefl spevfqhiwd 61 fleqpicsvq pidlnfvdep sedgatnkie ismdcirmqd sdlsdpmwpq ytnlgllnsm 121 dqqiqngsss tspyntdhaq nsvtapspya qpsstfdals pspaipsntd ypgphsfdvs 181 fqqsstaksa twtystelkk lycqiaktcp iqikvmtppp qgavirampv ykkaehvtev 241 vkrcpnhels refnegqiap pshlirvegn shaqyvedpi tgrqsvlvpy eppqvgteft 301 tvlynfmcns scvggmnrrp iliivtletr dgqvlgrrcf earicacpgr drkadedsir 361 kqqvsdstkn gdgtkrpfrq nthgiqmtsi kkrrspddel lylpvrgret yemllkikes 421 lelmqylpqh tietyrqqqq qqhqhllqkq tsiqspssyg nsspplnkmn smnklpsvsq 481 linpqqrnal tpttipdgmg anipmmgthm pmagdmngls ptqalpppls mpstshctpp 541 ppyptdcsiv sflarlgcss cldyfttqgl ttiyqiehys mddlaslkip eqfrhaiwkg 601 ildhrqlhef sspshllrtp ssastvsvgs setrgervid avrftlrqti sfpprdewnd 661 fnfdmdarrn kqqrikeege';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['epiophalmocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function glial\_cell\_2() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count26 = 0; count26 < 1; count26++) {

for (var count25 = 0; count25 < 4500; count25++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeeqdlsev elspvgseep rclspgsaps lgpdgggggs glraspgpge lgkvkkeqqd 61 geadddkfpv cireavsqvl sgydwtlvpm pvrvngasks kphvkrpmna fmvwaqaarr 121 kladqyphlh naelsktlgk lwrllnesdk rpfieeaerl rmqhkkdhpd ykyqprrrkn 181 gkaaqgeaec pggeaeqggt aaiqahyksa hldhrhpgeg spmsdgnpeh psgqshgppt 241 ppttpktelq sgkadpkrdg rsmgeggkph idfgnvdige ishevmsnme tfdvaeldqy 301 lppnghpghv ssysaagygl gsalavasgh sawiskppgv alptvsppgv dakaqvktet 361 agpqgpphyt dqpstsqiay tslslphygs afpsisrpqf dysdhqpsgp yyghsgqasg 421 lysafsymgp sqrplytais dpspsgpqsh spthweqpvy ttlsrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count28 = 0; count28 < 1; count28++) {

for (var count27 = 0; count27 < 4500; count27++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 maeeqdlsev elspvgseep rclspgsaps lgpdgggggs glraspgpge lgkvkkeqqd 61 geadddkfpv cireavsqvl sgydwtlvpm pvrvngasks kphvkrpmna fmvwaqaarr 121 kladqyphlh naelsktlgk lwrllnesdk rpfieeaerl rmqhkkdhpd ykyqprrrkn 181 gkaaqgeaec pggeaeqggt aaiqahyksa hldhrhpgeg spmsdgnpeh psgqshgppt 241 ppttpktelq sgkadpkrdg rsmgeggkph idfgnvdige ishevmsnme tfdvaeldqy 301 lppnghpghv ssysaagygl gsalavasgh sawiskppgv alptvsppgv dakaqvktet 361 agpqgpphyt dqpstsqiay tslslphygs afpsisrpqf dysdhqpsgp yyghsgqasg 421 lysafsymgp sqrplytais dpspsgpqsh spthweqpvy ttlsrp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count30 = 0; count30 < 1; count30++) {

for (var count29 = 0; count29 < 4500; count29++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmtakavdki pvtlsgfvhq lsdniypved laatsvtifp naelggpfdq mngvagdgmi 61 nidmtgekrs ldlpypssfa pvsaprnqtf tymgkfsidp qypgascype giinivsagi 121 lqgvtspast tasssvtsas pnplatgplg vctmsqtqpd ldhlyspppp pppysgcagd 181 lyqdpsafls aattstsssl ayppppsyps pkpatdpglf pmipdypgff psqcqrdlhg 241 tagpdrkpfp cpldtlrvpp pltplstirn ftlggpsagv tgpgasggse gprlpgsssa 301 aaaaaaaaay nphhlplrpi lrprkypnrp sktpvherpy pcpaegcdrr fsrsdeltrh 361 irihtghkpf qcricmrnfs rsdhltthir thtgekpfac dycgrkfars derkrhtkih 421 lrqkerkssa psasvpapst ascsggvqpg gtlcssnsss lgggplapcs srtrtp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count32 = 0; count32 < 1; count32++) {

for (var count31 = 0; count31 < 4500; count31++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmtakavdki pvtlsgfvhq lsdniypved laatsvtifp naelggpfdq mngvagdgmi 61 nidmtgekrs ldlpypssfa pvsaprnqtf tymgkfsidp qypgascype giinivsagi 121 lqgvtspast tasssvtsas pnplatgplg vctmsqtqpd ldhlyspppp pppysgcagd 181 lyqdpsafls aattstsssl ayppppsyps pkpatdpglf pmipdypgff psqcqrdlhg 241 tagpdrkpfp cpldtlrvpp pltplstirn ftlggpsagv tgpgasggse gprlpgsssa 301 aaaaaaaaay nphhlplrpi lrprkypnrp sktpvherpy pcpaegcdrr fsrsdeltrh 361 irihtghkpf qcricmrnfs rsdhltthir thtgekpfac dycgrkfars derkrhtkih 421 lrqkerkssa psasvpapst ascsggvqpg gtlcssnsss lgggplapcs srtrtp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count34 = 0; count34 < 1; count34++) {

for (var count33 = 0; count33 < 4500; count33++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmtakavdki pvtlsgfvhq lsdniypved laatsvtifp naelggpfdq mngvagdgmi 61 nidmtgekrs ldlpypssfa pvsaprnqtf tymgkfsidp qypgascype giinivsagi 121 lqgvtspast tasssvtsas pnplatgplg vctmsqtqpd ldhlyspppp pppysgcagd 181 lyqdpsafls aattstsssl ayppppsyps pkpatdpglf pmipdypgff psqcqrdlhg 241 tagpdrkpfp cpldtlrvpp pltplstirn ftlggpsagv tgpgasggse gprlpgsssa 301 aaaaaaaaay nphhlplrpi lrprkypnrp sktpvherpy pcpaegcdrr fsrsdeltrh 361 irihtghkpf qcricmrnfs rsdhltthir thtgekpfac dycgrkfars derkrhtkih 421 lrqkerkssa psasvpapst ascsggvqpg gtlcssnsss lgggplapcs srtrtp';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['glial cell 2', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function gustocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count36 = 0; count36 < 1; count36++) {

for (var count35 = 0; count35 < 4500; count35++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mlllarclll vlvssllvcs glacgpgrgf gkrrhpkklt playkqfipn vaektlgasg 61 ryegkisrns erfkeltpny npdiifkdee ntgadrlmtq rckdklnala isvmnqwpgv 121 klrvtegwde dghhseeslh yegravditt sdrdrskygm larlaveagf dwvyyeskah 181 ihcsvkaens vaaksggcfp gsatvhleqg gtklvkdlsp gdrvlaaddq grllysdflt 241 fldrddgakk vfyvietrep rerllltaah llfvaphnds atgepeassg sgppsggalg 301 pralfasrvr pgqrvyvvae rdgdrrllpa avhsvtlsee aagayaplta qgtilinrvl 361 ascyavieeh swahrafapf rlahallaal apartdrggd sgggdrgggg grvaltapga 421 adapgagata gihwysqlly qigtwlldse alhplgmavk ss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count38 = 0; count38 < 1; count38++) {

for (var count37 = 0; count37 < 4500; count37++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mlllarclll vlvssllvcs glacgpgrgf gkrrhpkklt playkqfipn vaektlgasg 61 ryegkisrns erfkeltpny npdiifkdee ntgadrlmtq rckdklnala isvmnqwpgv 121 klrvtegwde dghhseeslh yegravditt sdrdrskygm larlaveagf dwvyyeskah 181 ihcsvkaens vaaksggcfp gsatvhleqg gtklvkdlsp gdrvlaaddq grllysdflt 241 fldrddgakk vfyvietrep rerllltaah llfvaphnds atgepeassg sgppsggalg 301 pralfasrvr pgqrvyvvae rdgdrrllpa avhsvtlsee aagayaplta qgtilinrvl 361 ascyavieeh swahrafapf rlahallaal apartdrggd sgggdrgggg grvaltapga 421 adapgagata gihwysqlly qigtwlldse alhplgmavk ss';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count40 = 0; count40 < 1; count40++) {

for (var count39 = 0; count39 < 4500; count39++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count42 = 0; count42 < 1; count42++) {

for (var count41 = 0; count41 < 4500; count41++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count44 = 0; count44 < 1; count44++) {

for (var count43 = 0; count43 < 4500; count43++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdtsrlgvll slpvllqlat ggssprsgvl lrgcpthchc epdgrmllrv dcsdlglsel 61 psnlsvftsy ldlsmnnisq llpnplpslr fleelrlagn altyipkgaf tglyslkvlm 121 lqnnqlrhvp tealqnlrsl qslrldanhi syvppscfsg lhslrhlwld dnalteipvq 181 afrslsalqa mtlalnkihh ipdyafgnls slvvlhlhnn rihslgkkcf dglhsletld 241 lnynnldefp tairtlsnlk elgfhsnnir sipekafvgn pslitihfyd npiqfvgrsa 301 fqhlpelrtl tlngasqite fpdltgtanl esltltgaqi sslpqtvcnq lpnlqvldls 361 ynlledlpsf svcqklqkid lrhneiyeik vdtfqqllsl rslnlawnki aiihpnafst 421 lpslikldls snllssfpit glhglthlkl tgnhalqsli ssenfpelkv iempyayqcc 481 afgvcenayk isnqwnkgdn ssmddlhkkd agmfqaqder dledflldfe edlkalhsvq 541 cspspgpfkp cehlldgwli rigvwtiavl altcnalvts tvfrsplyis piklligvia 601 avnmltgvss avlagvdaft fgsfarhgaw wengvgchvi gflsifases svflltlaal 661 ergfsvkysa kfetkapfss lkviillcal laltmaavpl lggskygasp lclplpfgep 721 stmgymvali llnslcflmm tiaytklycn ldkgdleniw dcsmvkhial llftncilnc 781 pvaflsfssl inltfispev ikfillvvvp lpaclnplly ilfnphfked lvslrkqtyv 841 wtrskhpslm sinsddvekq scdstqalvt ftsssitydl ppssvpspay pvteschlss 901 vafvpcl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count46 = 0; count46 < 1; count46++) {

for (var count45 = 0; count45 < 4500; count45++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msrrkqgnpq hlsqrelitp eadhveaail eedegleiee psglglmvgg pdpdlltcgq 61 cqmnfplgdi lvfiehkrkq cggslgacyd kaldkdsppp ssrselrkvs epveigiqvt 121 pdeddhllsp tkgicpkqen iagpcrpaql pavapiaass hphssvitsp lralgalppc 181 lplpccsarp vsgdgtqgeg qteapfgcqc qlsgkdepss yicttckqpf nsawfllqha 241 qnthgfriyl epgpassslt prltippplg peavaqsplm nflgdsnpfn llrmtgpilr 301 dhpgfgegrl pgtpplfspp prhhldphrl saeemglvaq hpsafdrvmr lnpmaidspa 361 mdfsrrlrel agnsstpppv spgrgnpmhr llnpfqpspk spflstpplp pmppggtppp 421 qppaksksce fcgktfkfqs nlivhrrsht gekpykcqlc dhacsqaskl krhmkthmhk 481 agslagrsdd glsaasspep gtselagegl kaadgdfrhh esdpslghep eeedeeeeee 541 eeelllenes rpessfsmds elsrnrengg ggvpgvpgag ggaakalade kalvlgkvme 601 nvglgalpqy gelladkqkr gaflkraagg gdagddddag gcgdagagga vngrgggfap 661 gtepfpglfp rkpaplpspg lnsaakrikv ekdlelppaa lipsenvysq wlvgyaasrh 721 fmkdpflgft darqspfats sehssengsl rfstppgdll dgglsgrsgt asggstphlg 781 gpgpgrpssk egrrsdtcey cgkvfkncsn ltvhrrshtg erpykcelcn yacaqssklt 841 rhmkthgqig kevyrcdicq mpfsvystle khmkkwhgeh lltndvkieq aers';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['gustocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function hypothalamocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count48 = 0; count48 < 1; count48++) {

for (var count47 = 0; count47 < 4500; count47++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvfrspldly sshfllpnfa dshhrsilla ssgggngagg gggagggsgg gngaggggag 61 gagggggggs rappeelsmf qlptlnfspe qvasvcetle etgdierlgr flwslpvapg 121 aceainkhes ilraravvaf htgnfrdlyh ilenhkftke shgklqamwl eahyqeaekl 181 rgrplgpvdk yrvrkkfplp rtiwdgeqkt hcfkertrsl lrewylqdpy pnpskkrela 241 qatgltptqv gnwfknrrqr draaaaknrl qhqaigpsgm rslaepgcpt hgsaespsta 301 aspttsvssl teradtgtsi lsvtssdsec dv';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count50 = 0; count50 < 1; count50++) {

for (var count49 = 0; count49 < 4500; count49++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfqlpilnfs pqqvagvcet leesgdverl grflwslpva paacealnkn esvlraraiv 61 afhggnyrel yhilenhkft keshaklqal wleahyqeae klrgrplgpv dkyrvrkkfp 121 lprtiwdgeq kthcfkertr hllrewylqd pypnpskkre laqatgltpt qvgnwfknrr 181 qrdraaaakn rlqqqvlsqg sgralraegd gtpevlgvat spaaslsska atsaisitss 241 dsecdi';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count52 = 0; count52 < 1; count52++) {

for (var count51 = 0; count51 < 4500; count51++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mvhcagckrp ildrfllnvl drawhvkcvq cceckcnlte kcfsregkly ckndffrcfg 61 tkcagcaqgi spsdlvrrar skvfhlncft cmmcnkqlst geelyiiden kfvckedyls 121 nssvakensl hsattgsdps lspdsqdpsq ddakdsesan vsdkeagsne nddqnlgakr 181 rgprttikak qletlkaafa atpkptrhir eqlaqetgln mrviqvwfqn rrskerrmkq 241 lsalgarrha ffrsprrmrp lvdrlepgel ipngpfsfyg dyqseyygpg gnydffpqgp 301 pssqaqtpvd lpfvpssgps gtplgglehp lpghhpssea qrftdilahp pgdspspeps 361 lpgplhsmsa evfgpsppfs slsvnggasy gnhlshppem neaavw';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count54 = 0; count54 < 1; count54++) {

for (var count53 = 0; count53 < 4500; count53++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlfhslsgpe vhgvidemdr rakseapais saidrgdtet tmpsissdra alcagcggki 61 sdryyllavd kqwhmrclkc cecklnlese ltcfskdgsi yckedyyrrf svqrcarchl 121 gisasemvmr ardlvyhlnc ftcttcnkml ttgdhfgmkd slvycrlhfe allqgeypah 181 fnhadvaaaa aaaaaaksag lgaaganplg lpyyngvgtv qkgrprkrks pgpgadlaay 241 naalscnend aehldrdqpy pssqktkrmr tsfkhhqlrt mksyfainhn pdakdlkqla 301 qktgltkrvl qvwfqnarak frrnllrqen tgvdkstdaa lqtgtpsgpa selsnaslsp 361 sstpttltdl tsptlptvts vltsvpgnle ghephspsqt tltnlf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count56 = 0; count56 < 1; count56++) {

for (var count55 = 0; count55 < 4500; count55++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mlfhslsgpe vhgvidemdr rakseapais saidrgdtet tmpsissdra alcagcggki 61 sdryyllavd kqwhmrclkc cecklnlese ltcfskdgsi yckedyyrrf svqrcarchl 121 gisasemvmr ardlvyhlnc ftcttcnkml ttgdhfgmkd slvycrlhfe allqgeypah 181 fnhadvaaaa aaaaaaksag lgaaganplg lpyyngvgtv qkgrprkrks pgpgadlaay 241 naalscnend aehldrdqpy pssqktkrmr tsfkhhqlrt mksyfainhn pdakdlkqla 301 qktgltkrvl qvwfqnarak frrnllrqen tgvdkstdaa lqtgtpsgpa selsnaslsp 361 sstpttltdl tsptlptvts vltsvpgnle ghephspsqt tltnlf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['hypothalamocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function neuroblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count58 = 0; count58 < 1; count58++) {

for (var count57 = 0; count57 < 4500; count57++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mparletcis dldcasssgs dlsgfltdee dcarlqqaas asgppaparr gapnisrase 61 vpgaqddeqe rrrrrgrtrv rseallhslr rsrrvkandr ernrmhnlna aldalrsvlp 121 sfpddtkltk ietlrfayny iwalaetlrl adqglpggga rerllppqcv pclpgppspa 181 sdaeswgsga aaasplsdps spaasedfty rpgdpvfsfp slpkdllhtt pcfipyh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count60 = 0; count60 < 1; count60++) {

for (var count59 = 0; count59 < 4500; count59++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mfvksetlel keeedvlvll gsaspalaal tplsssadee eeeepgasgg arrqrgaeag 61 qgarggvaag aegcrparll glvhdckrrp sraravsrga ktaetvqrik ktrrlkannr 121 ernrmhnlna aldalrevlp tfpedakltk ietlrfahny iwaltetlrl adhcgggggg 181 lpgalfseav llspggasaa lsssgdspsp astwsctnsp apsssvssns tspysctlsp 241 aspagsdmdy wqppppdkhr yaphlpiard ci';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count62 = 0; count62 < 1; count62++) {

for (var count61 = 0; count61 < 4500; count61++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mtpqpsgapt vqvtreters fprasedevt cptsappspt rtrgncaeae eggcrgaprk 61 lrarrggrsr pkselalskq rrsrrkkand rernrmhnln saldalrgvl ptfpddaklt 121 kietlrfahn yiwaltqtlr iadhslyale ppaphcgelg spggspgdwg slyspvsqag 181 slspaaslee rpgllgatfs aclspgslaf sdfl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count64 = 0; count64 < 1; count64++) {

for (var count63 = 0; count63 < 4500; count63++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtksysesgl mgepqpqgpp swtdeclssq deeheadkke ddletmnaee dslrnggeee 61 dededleeee eeeeedddqk pkrrgpkkkk mtkarlerfk lrrmkanare rnrmhglnaa 121 ldnlrkvvpc ysktqklski etlrlaknyi walseilrsg kspdlvsfvq tlckglsqpt 181 tnlvagclql nprtflpeqn qdmpphlpta sasfpvhpys yqspglpspp ygtmdsshvf 241 hvkppphays aalepffesp ltdctspsfd gplspplsin gnfsfkheps aefeknyaft 301 mhypaatlag aqshgsifsg taaprceipi dnimsfdshs hhervmsaql naifhd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count66 = 0; count66 < 1; count66++) {

for (var count65 = 0; count65 < 4500; count65++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mltrlfsepg llsdvpkfas wgdgeddepr sdkgdapppp ppapgpgapg paraakpvpl 61 rgeegteatl aevkeegelg geeeeeeeee egldeaeger pkkrgpkkrk mtkarlersk 121 lrrqkanare rnrmhdlnaa ldnlrkvvpc ysktqklski etlrlaknyi walseilrsg 181 krpdlvsyvq tlckglsqpt tnlvagclql nsrnflteqg adgagrfhgs ggpfamhpyp 241 ypcsrlagaq cqaagglggg aahalrthgy caayetlyaa aggggaspdy nsseyegpls 301 pplclngnfs lkqdsspdhe ksyhysmhys alpgsrptgh glvfgssavr ggvhsenlls 361 ydmhlhhdrg pmyeelnaff hn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['neuroblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function olfactocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count68 = 0; count68 < 1; count68++) {

for (var count67 = 0; count67 < 4500; count67++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqnshsgvnq lggvfvngrp lpdstrqkiv elahsgarpc disrilqvsn gcvskilgry 61 yetgsirpra iggskprvat pevvskiaqy krecpsifaw eirdrllseg vctndnipsv 121 ssinrvlrnl asekqqmgad gmydklrmln gqtgswgtrp gwypgtsvpg qptqdgcqqq 181 egggentnsi ssngedsdea qmrlqlkrkl qrnrtsftqe qiealekefe rthypdvfar 241 erlaakidlp eariqvwfsn rrakwrreek lrnqrrqasn tpshipisss fstsvyqpip 301 qpttpvssft sgsmlgrtdt altntysalp pmpsftmann lpmqppvpsq tssyscmlpt 361 spsvngrsyd tytpphmqth mnsqpmgtsg ttstglispg vsvpvqvpgs epdmsqywpr 421 lq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count70 = 0; count70 < 1; count70++) {

for (var count69 = 0; count69 < 4500; count69++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqnshsgvnq lggvfvngrp lpdstrqkiv elahsgarpc disrilqvsn gcvskilgry 61 yetgsirpra iggskprvat pevvskiaqy krecpsifaw eirdrllseg vctndnipsv 121 ssinrvlrnl asekqqmgad gmydklrmln gqtgswgtrp gwypgtsvpg qptqdgcqqq 181 egggentnsi ssngedsdea qmrlqlkrkl qrnrtsftqe qiealekefe rthypdvfar 241 erlaakidlp eariqvwfsn rrakwrreek lrnqrrqasn tpshipisss fstsvyqpip 301 qpttpvssft sgsmlgrtdt altntysalp pmpsftmann lpmqppvpsq tssyscmlpt 361 spsvngrsyd tytpphmqth mnsqpmgtsg ttstglispg vsvpvqvpgs epdmsqywpr 421 lq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count72 = 0; count72 < 1; count72++) {

for (var count71 = 0; count71 < 4500; count71++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdmhckadpf samhpghggv nqlggvfvng rplpdvvrqr ivelahqgvr pcdisrqlrv 61 shgcvskilg ryyetgsikp gviggskpkv atpkvvdkia eykrqnptmf aweirdrlla 121 egicdndtvp svssinriir tkvqqpfhpt pdgagtgvta pghtivpsta sppvssasnd 181 pvgsysingi lgiprsngek rkrdevevyt dpahirgggg lhlvwtlrdv segsvpngds 241 qsgvdslrkh lradtftqqq lealdrvfer psypdvfqas ehikseqgne yslpaltpgl 301 devksslsas tnpelgsnvs gtqtypvvtg rdmasttlpg ypphvpptgq gsyptstlag 361 mvpgsefsgn pyshpqytay neawrfsnpa llsspyyysa aprgsapaaa aaaydrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count74 = 0; count74 < 1; count74++) {

for (var count73 = 0; count73 < 4500; count73++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mdmhckadpf samhpghggv nqlggvfvng rplpdvvrqr ivelahqgvr pcdisrqlrv 61 shgcvskilg ryyetgsikp gviggskpkv atpkvvdkia eykrqnptmf aweirdrlla 121 egicdndtvp svssinriir tkvqqpfhpt pdgagtgvta pghtivpsta sppvssasnd 181 pvgsysingi lgiprsngek rkrdevevyt dpahirgggg lhlvwtlrdv segsvpngds 241 qsgvdslrkh lradtftqqq lealdrvfer psypdvfqas ehikseqgne yslpaltpgl 301 devksslsas tnpelgsnvs gtqtypvvtg rdmasttlpg ypphvpptgq gsyptstlag 361 mvpgsefsgn pyshpqytay neawrfsnpa llsspyyysa aprgsapaaa aaaydrh';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count76 = 0; count76 < 1; count76++) {

for (var count75 = 0; count75 < 4500; count75++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmdnrgnssl pdklpifpds arlpltrsfy lepmvtfhvh peapvsspys eelprlpfps 61 dslilgnyse pcpfsfpmpy pnyrgceysy gpaftrkrne rerqrvkcvn egyaqlrhhl 121 peeylekrls kvetlraaik yinylqslly pdkaetknnp gkvssmiatt shhadpmfri 181 v';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count78 = 0; count78 < 1; count78++) {

for (var count77 = 0; count77 < 4500; count77++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmdnrgnssl pdklpifpds arlpltrsfy lepmvtfhvh peapvsspys eelprlpfps 61 dslilgnyse pcpfsfpmpy pnyrgceysy gpaftrkrne rerqrvkcvn egyaqlrhhl 121 peeylekrls kvetlraaik yinylqslly pdkaetknnp gkvssmiatt shhadpmfri 181 v';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['olfactocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function ophalmoblast() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count80 = 0; count80 < 1; count80++) {

for (var count79 = 0; count79 < 4500; count79++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmaymnpgph ysvnalalsg psvdlmhqav pypsaprkqr rerttftrsq leelealfak 61 tqypdvyare evalkinlpe srvqvwfknr rakcrqqrqq qkqqqqppgg qakarpakrk 121 agtsprpstd vcpdplgisd syspplpgps gspttavatv siwspasesp lpeaqraglv 181 asgpsltsap yamtyapasa fcsspsaygs pssyfsgldp ylspmvpqlg gpalsplsgp 241 svgpslaqsp tslsgqsyga yspvdslefk dptgtwkfty npmdpldykd qsawkfqil';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count82 = 0; count82 < 1; count82++) {

for (var count81 = 0; count81 < 4500; count81++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmaymnpgph ysvnalalsg psvdlmhqav pypsaprkqr rerttftrsq leelealfak 61 tqypdvyare evalkinlpe srvqvwfknr rakcrqqrqq qkqqqqppgg qakarpakrk 121 agtsprpstd vcpdplgisd syspplpgps gspttavatv siwspasesp lpeaqraglv 181 asgpsltsap yamtyapasa fcsspsaygs pssyfsgldp ylspmvpqlg gpalsplsgp 241 svgpslaqsp tslsgqsyga yspvdslefk dptgtwkfty npmdpldykd qsawkfqil';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count84 = 0; count84 < 1; count84++) {

for (var count83 = 0; count83 < 4500; count83++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhlpgcapam adgsfslagh llrspggsts rlhsieailg ftkddgilgt fpaergarga 61 kerdrrlgar pacpkapeeg sepspppapa papeyeaprp ycpkepgear pspglpvgpa 121 tgeaklseee qpkkkhrrnr ttfttyqlhe lerafekshy pdvysreela gkvnlpevrv 181 qvwfqnrrak wrrqeklevs smklqdspll sfsrsppsat lsplgagpgs gggpaggalp 241 leswlgpplp gggatalqsl pgfgppaqsl pasytppppp ppflnspplg pglqplappp 301 psypcgpgfg dkfpldeadp rnssiaalrl kakehiqaig kpwqal';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count86 = 0; count86 < 1; count86++) {

for (var count85 = 0; count85 < 4500; count85++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mhlpgcapam adgsfslagh llrspggsts rlhsieailg ftkddgilgt fpaergarga 61 kerdrrlgar pacpkapeeg sepspppapa papeyeaprp ycpkepgear pspglpvgpa 121 tgeaklseee qpkkkhrrnr ttfttyqlhe lerafekshy pdvysreela gkvnlpevrv 181 qvwfqnrrak wrrqeklevs smklqdspll sfsrsppsat lsplgagpgs gggpaggalp 241 leswlgpplp gggatalqsl pgfgppaqsl pasytppppp ppflnspplg pglqplappp 301 psypcgpgfg dkfpldeadp rnssiaalrl kakehiqaig kpwqal';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count88 = 0; count88 < 1; count88++) {

for (var count87 = 0; count87 < 4500; count87++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtksysesgl mgepqpqgpp swtdeclssq deeheadkke ddletmnaee dslrnggeee 61 dededleeee eeeeedddqk pkrrgpkkkk mtkarlerfk lrrmkanare rnrmhglnaa 121 ldnlrkvvpc ysktqklski etlrlaknyi walseilrsg kspdlvsfvq tlckglsqpt 181 tnlvagclql nprtflpeqn qdmpphlpta sasfpvhpys yqspglpspp ygtmdsshvf 241 hvkppphays aalepffesp ltdctspsfd gplspplsin gnfsfkheps aefeknyaft 301 mhypaatlag aqshgsifsg taaprceipi dnimsfdshs hhervmsaql naifhd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count90 = 0; count90 < 1; count90++) {

for (var count89 = 0; count89 < 4500; count89++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mtksysesgl mgepqpqgpp swtdeclssq deeheadkke ddletmnaee dslrnggeee 61 dededleeee eeeeedddqk pkrrgpkkkk mtkarlerfk lrrmkanare rnrmhglnaa 121 ldnlrkvvpc ysktqklski etlrlaknyi walseilrsg kspdlvsfvq tlckglsqpt 181 tnlvagclql nprtflpeqn qdmpphlpta sasfpvhpys yqspglpspp ygtmdsshvf 241 hvkppphays aalepffesp ltdctspsfd gplspplsin gnfsfkheps aefeknyaft 301 mhypaatlag aqshgsifsg taaprceipi dnimsfdshs hhervmsaql naifhd';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['ophalmoblast', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function ophalmocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count92 = 0; count92 < 1; count92++) {

for (var count91 = 0; count91 < 4500; count91++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmsylkqppy avnglsltts gmdllhpsvg ypgpwascpa atprkqrrer ttftraqldv 61 lealfaktry pdifmreeva lkinlpesrv qvwfknrrak crqqqqqqqn ggqnkvrpak 121 kktsparevs sesgtsgqft ppsstsvpti asssapvsiw spasisplsd plstssscmq 181 rsypmtytqa sgysqgyags tsyfggmdcg syltpmhhql pgpgatlspm gtnavtshln 241 qspaslstqg ygasslgfns ttdcldykdq taswklnfna dcldykdqts swkfqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count94 = 0; count94 < 1; count94++) {

for (var count93 = 0; count93 < 4500; count93++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mmsylkqppy avnglsltts gmdllhpsvg ypgpwascpa atprkqrrer ttftraqldv 61 lealfaktry pdifmreeva lkinlpesrv qvwfknrrak crqqqqqqqn ggqnkvrpak 121 kktsparevs sesgtsgqft ppsstsvpti asssapvsiw spasisplsd plstssscmq 181 rsypmtytqa sgysqgyags tsyfggmdcg syltpmhhql pgpgatlspm gtnavtshln 241 qspaslstqg ygasslgfns ttdcldykdq taswklnfna dcldykdqts swkfqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count96 = 0; count96 < 1; count96++) {

for (var count95 = 0; count95 < 4500; count95++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqnshsgvnq lggvfvngrp lpdstrqkiv elahsgarpc disrilqvsn gcvskilgry 61 yetgsirpra iggskprvat pevvskiaqy krecpsifaw eirdrllseg vctndnipsv 121 ssinrvlrnl asekqqmgad gmydklrmln gqtgswgtrp gwypgtsvpg qptqdgcqqq 181 egggentnsi ssngedsdea qmrlqlkrkl qrnrtsftqe qiealekefe rthypdvfar 241 erlaakidlp eariqvwfsn rrakwrreek lrnqrrqasn tpshipisss fstsvyqpip 301 qpttpvssft sgsmlgrtdt altntysalp pmpsftmann lpmqppvpsq tssyscmlpt 361 spsvngrsyd tytpphmqth mnsqpmgtsg ttstglispg vsvpvqvpgs epdmsqywpr 421 lq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count98 = 0; count98 < 1; count98++) {

for (var count97 = 0; count97 < 4500; count97++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqnshsgvnq lggvfvngrp lpdstrqkiv elahsgarpc disrilqvsn gcvskilgry 61 yetgsirpra iggskprvat pevvskiaqy krecpsifaw eirdrllseg vctndnipsv 121 ssinrvlrnl asekqqmgad gmydklrmln gqtgswgtrp gwypgtsvpg qptqdgcqqq 181 egggentnsi ssngedsdea qmrlqlkrkl qrnrtsftqe qiealekefe rthypdvfar 241 erlaakidlp eariqvwfsn rrakwrreek lrnqrrqasn tpshipisss fstsvyqpip 301 qpttpvssft sgsmlgrtdt altntysalp pmpsftmann lpmqppvpsq tssyscmlpt 361 spsvngrsyd tytpphmqth mnsqpmgtsg ttstglispg vsvpvqvpgs epdmsqywpr 421 lq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count100 = 0; count100 < 1; count100++) {

for (var count99 = 0; count99 < 4500; count99++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqsesgivpd fevgeefhee pktyyelksq plkssssaeh pgaskppiss ssmtsrillr 61 qqlmreqmqe qerreqqqkl qaaqfmqqrv pvsqtpainv svpttlpsat qvpmevlkvq 121 thlenptkyh iqqaqrqqvk qylsttlank hanqvlslpc pnqpgdhvmp pvpgssapns 181 pmamltlnsn cekegfykfe eqnraesecp gmnthsrasc mqmddviddi islessynee 241 ilglmdpalq mantlpvsgn lidlygnqgl pppgltisns cpanlpnikr eltesearal 301 akerqkkdnh nlierrrrfn indrikelgt lipksndpdm rwnkgtilka svdyirklqr 361 eqqrakelen rqkklehanr hlllriqele mqarahglsl ipstglcspd lvnriikqep 421 vlencsqdll qhhadltctt tldltdgtit fnnnlgtgte anqaysvptk mgskledilm 481 ddtlspvgvt dpllssvspg asktssrrss msmeetehtc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count102 = 0; count102 < 1; count102++) {

for (var count101 = 0; count101 < 4500; count101++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mqsesgivpd fevgeefhee pktyyelksq plkssssaeh pgaskppiss ssmtsrillr 61 qqlmreqmqe qerreqqqkl qaaqfmqqrv pvsqtpainv svpttlpsat qvpmevlkvq 121 thlenptkyh iqqaqrqqvk qylsttlank hanqvlslpc pnqpgdhvmp pvpgssapns 181 pmamltlnsn cekegfykfe eqnraesecp gmnthsrasc mqmddviddi islessynee 241 ilglmdpalq mantlpvsgn lidlygnqgl pppgltisns cpanlpnikr eltesearal 301 akerqkkdnh nlierrrrfn indrikelgt lipksndpdm rwnkgtilka svdyirklqr 361 eqqrakelen rqkklehanr hlllriqele mqarahglsl ipstglcspd lvnriikqep 421 vlencsqdll qhhadltctt tldltdgtit fnnnlgtgte anqaysvptk mgskledilm 481 ddtlspvgvt dpllssvspg asktssrrss msmeetehtc';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['ophalmocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function otocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count104 = 0; count104 < 1; count104++) {

for (var count103 = 0; count103 < 4500; count103++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msrllhaeew aevkelgdhh rqpqphhlpq pppppqppat lqarehpvyp pelslldstd 61 prawlaptlq gictaraaqy llhspelgas eaaaprdevd grgelvrrss ggassskspg 121 pvkvreqlck lkggvvvdel gcsrqrapss kqvngvqkqr rlaanarerr rmhglnhafd 181 qlrnvipsfn ndkklskyet lqmaqiyina lsellqtpsg geqpppppas cksdhhhlrt 241 aasyeggagn ataagaqqas ggsqrptppg scrtrfsapa saggysvqld alhfstfeds 301 altammaqkn lspslpgsil qpvqeenskt sprshrsdge fsphshysds deas';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count106 = 0; count106 < 1; count106++) {

for (var count105 = 0; count105 < 4500; count105++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msrllhaeew aevkelgdhh rqpqphhlpq pppppqppat lqarehpvyp pelslldstd 61 prawlaptlq gictaraaqy llhspelgas eaaaprdevd grgelvrrss ggassskspg 121 pvkvreqlck lkggvvvdel gcsrqrapss kqvngvqkqr rlaanarerr rmhglnhafd 181 qlrnvipsfn ndkklskyet lqmaqiyina lsellqtpsg geqpppppas cksdhhhlrt 241 aasyeggagn ataagaqqas ggsqrptppg scrtrfsapa saggysvqld alhfstfeds 301 altammaqkn lspslpgsil qpvqeenskt sprshrsdge fsphshysds deas';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count108 = 0; count108 < 1; count108++) {

for (var count107 = 0; count107 < 4500; count107++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count110 = 0; count110 < 1; count110++) {

for (var count109 = 0; count109 < 4500; count109++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count112 = 0; count112 < 1; count112++) {

for (var count111 = 0; count111 < 4500; count111++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msmlpsfgft qeqvacvcev lqqggnlerl grflwslpac dhlhknesvl kakavvafhr 61 gnfrelykil eshqfsphnh pklqqlwlka hyveaeklrg rplgavgkyr vrrkfplprt 121 iwdgeetsyc fkeksrgvlr ewyahnpyps prekrelaea tgltttqvsn wfknrrqrdr 181 aaeakerent ennnsssnkq nqlspleggk plmssseeef sppqspdqns vlllqgnmgh 241 arssnyslpg ltasqpshgl qthqhqlqds llgpltsslv dlgs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count114 = 0; count114 < 1; count114++) {

for (var count113 = 0; count113 < 4500; count113++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 msmlpsfgft qeqvacvcev lqqggnlerl grflwslpac dhlhknesvl kakavvafhr 61 gnfrelykil eshqfsphnh pklqqlwlka hyveaeklrg rplgavgkyr vrrkfplprt 121 iwdgeetsyc fkeksrgvlr ewyahnpyps prekrelaea tgltttqvsn wfknrrqrdr 181 aaeakerent ennnsssnkq nqlspleggk plmssseeef sppqspdqns vlllqgnmgh 241 arssnyslpg ltasqpshgl qthqhqlqds llgpltsslv dlgs';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['otocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function pineocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count116 = 0; count116 < 1; count116++) {

for (var count115 = 0; count115 < 4500; count115++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmsyikqphy avngltltgp gmdllhsavg yqttprkqrr erttftraql dileslfakt 61 rypdifmree valkinlpes rvqvwfknrr akcrqqqqqs sgqakarptk kktsptrdts 121 setngnghys ptppgaagtp ssisnatvsi wspasispip dplssstasc mqrsaaypmt 181 ysqaqgynqs ytgsssyfsg ldcgaylspm hpqlsapgaa lsplgtptmg ghlgqspasl 241 saqgyssagl gfntvdcldy kdqstswkln fsasdcldyk dqtswkfqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count118 = 0; count118 < 1; count118++) {

for (var count117 = 0; count117 < 4500; count117++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmsylkqppy avnglsltts gmdllhpsvg ypgpwascpa atprkqrrer ttftraqldv 61 lealfaktry pdifmreeva lkinlpesrv qvwfknrrak crqqqqqqqn ggqnkvrpak 121 kktsparevs sesgtsgqft ppsstsvpti asssapvsiw spasisplsd plstssscmq 181 rsypmtytqa sgysqgyags tsyfggmdcg syltpmhhql pgpgatlspm gtnavtshln 241 qspaslstqg ygasslgfns ttdcldykdq taswklnfna dcldykdqts swkfqvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count120 = 0; count120 < 1; count120++) {

for (var count119 = 0; count119 < 4500; count119++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mnlnftsplh passqrptsf fiedillhkp kplrevapdh fasslasrvp lldygyplmp 61 tptllaphah hplhkgdhhh pyflttsgmp vpalfphpqh aelpgkhcrr rkartvfsds 121 qlsglekrfe iqrylstper velatalsls etqvktwfqn rrmkhkkqlr ksqdepkapd 181 gpespegspr gseaataaea rlslpagpfv ltepedevdi gdegelgsgp hvl';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count122 = 0; count122 < 1; count122++) {

for (var count121 = 0; count121 < 4500; count121++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mqnshsgvnq lggvfvngrp lpdstrqkiv elahsgarpc disrilqvsn gcvskilgry 61 yetgsirpra iggskprvat pevvskiaqy krecpsifaw eirdrllseg vctndnipsv 121 ssinrvlrnl asekqqmgad gmydklrmln gqtgswgtrp gwypgtsvpg qptqdgcqqq 181 egggentnsi ssngedsdea qmrlqlkrkl qrnrtsftqe qiealekefe rthypdvfar 241 erlaakidlp eariqvwfsn rrakwrreek lrnqrrqasn tpshipisss fstsvyqpip 301 qpttpvssft sgsmlgrtdt altntysalp pmpsftmann lpmqppvpsq tssyscmlpt 361 spsvngrsyd tytpphmqth mnsqpmgtsg ttstglispg vsvpvqvpgs epdmsqywpr 421 lq';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count124 = 0; count124 < 1; count124++) {

for (var count123 = 0; count123 < 4500; count123++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 meivgcraed nscpfrppam lfhgisgghi qgimeemerr sktearlakg aqlngrdagm 61 pplspekpal cagcggkisd ryyllavdkq whlrclkcce cklaleselt cfakdgsiyc 121 kedyyrrfsv qrcarchlgi sasemvmrar dsvyhlscft cstcnktltt gdhfgmkdsl 181 vycrahfetl lqgeyppqls ytelaaksgg lalpyfngtg tvqkgrprkr kspalgvdiv 241 nynsgcnene adhldrdqqp yppsqktkrm rtsfkhhqlr tmksyfainh npdakdlkql 301 aqktgltkrv lqvwfqnara kfrrnllrqe nggvdkadgt slpappsads galtppgtat 361 tltdltnpti tvvtsvtsnm dshesgspsq ttltnlf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['pineocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function pitocyte() {

set\_cell\_rotation();

loc1 = [spawnerz, spawnery, spawnerx];

loc2 = [spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29];

make = true;

screen\_molecules();

screen\_cells();

if (make == true) {

default\_cell\_body();

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.631013891e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count126 = 0; count126 < 1; count126++) {

for (var count125 = 0; count125 < 4500; count125++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mynmmetelk ppgpqqtsgg gggnstaaaa ggnqknspdr vkrpmnafmv wsrgqrrkma 61 qenpkmhnse iskrlgaewk llsetekrpf ideakrlral hmkehpdyky rprrktktlm 121 kkdkytlpgg llapggnsma sgvgvgaglg agvnqrmdsy ahmngwsngs ysmmqdqlgy 181 pqhpglnahg aaqmqpmhry dvsalqynsm tssqtymngs ptysmsysqq gtpgmalgsm 241 gsvvkseass sppvvtsssh srapcqagdl rdmismylpg aevpepaaps rlhmsqhyqs 301 gpvpgtaing tlplshm';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.692893165e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count128 = 0; count128 < 1; count128++) {

for (var count127 = 0; count127 < 4500; count127++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 meaerrrqae kpkkgrvgsn llperhpatg tptttvdssa ppcrrlpgag ggrsrfspqg 61 gqrgrphsrr rhrttfspvq leqlesafgr nqypdiware slardtglse ariqvwfqnr 121 rakqrkqers llqplahlsp aafssflpes tacpysyaap pppvtcfphp yshalpsqps 181 tggafalshq sedwyptlhp apaghlpcpp pppmlplsle pskswn';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.754772438e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count130 = 0; count130 < 1; count130++) {

for (var count129 = 0; count129 < 4500; count129++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mscqaftsad tfiplnsdas atlplimhhs aaeclpvsnh atnvmstatg lhysvpschy 61 gnqpstygvm agsltpclyk fpdhtlshgf ppihqpllae dptaadfkqe lrrksklvee 121 pidmdspeir elekfanefk vrriklgytq tnvgealaav hgsefsqtti crfenlqlsf 181 knacklkail skwleeaeqv galynekvga nerkrkrrtt isiaakdale rhfgeqnkps 241 sqeimrmaee lnlekevvrv wfcnrrqrek rvktslnqsl fsiskehlec r';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.816651712e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count132 = 0; count132 < 1; count132++) {

for (var count131 = 0; count131 < 4500; count131++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = '1 mspslqegaq lgenkpstcs fsierilgld qkkdcvplmk phrpwadtcs ssgkdgnlcl 61 hvpnppsgis fpsvvdhpmp eeraskyeny fsaserlslk relswyrgrr prtaftqnqi 121 evlenvfrvn cypgidired laqklnleed riqiwfqnrr aklkrshres qflmakknfn 181 tnlle';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.847591349e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count134 = 0; count134 < 1; count134++) {

for (var count133 = 0; count133 < 4500; count133++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mlletglerd rarpgaaavc tlggtreipl cagcdqhild rfilkaldrh whskclkcsd 61 chtplaercf srgesvyckd dffkrfgtkc aacqlgippt qvvrraqdfv yhlhcfacvv 121 ckrqlatgde fylmedsrlv ckadyetakq reaeatakrp rttitakqle tlksayntsp 181 kparhvreql ssetgldmrv vqvwfqnrra kekrlkkdag rqrwgqyfrn mkrsrggsks 241 dkdsvqegqd sdaevsfpde pslaemgpan glygslgept qalgrpsgal gnfslehggl 301 agpeqyrelr pgspygvpps paapqslpgp qpllsslvyp dtslglvpsg apggpppmrv 361 lagngpssdl stgssggypd fpaspaswld evdhaqf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

if (unused\_variable == 5) {

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = spawnery - 5.859967203e+29 \* 1;

cursor1z = spawnerz + 3.15584295e+28 \* 1;

for (var count136 = 0; count136 < 1; count136++) {

for (var count135 = 0; count135 < 4500; count135++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

peptidesequence = ' 1 mmqsatvpae gavkglpeml gvpmqqipqc agcnqhildk filkvldrhw hssclkcadc 61 qmqladrcfs ragsvycked ffkrfgtkct acqqgipptq vvrkaqdfvy hlhcfaciic 121 nrqlatgdef ylmedgrlvc kedyetakqn ddseagakrp rttitakqle tlknayknsp 181 kparhvreql ssetgldmrv vqvwfqnrra kekrlkkdag rhrwgqfyks vkrsrgsskq 241 ekessaedcg vsdselsfre dqilselght nriygnvgdv tggqlmngsf smdgtgqsyq 301 dlrdgspygi pqspssissl pshapllngl dytvdsnlgi iahagqgvsq tlramaggpt 361 sdistgssvg ypdfptspgs wldemdhppf';

protein();

}

cursor1x = cursor1x - 6.187927353e+25 \* 2;

}

cursor1x = spawnerx - 3.15584295e+28 \* 1;

cursor1y = cursor1y - 6.187927353e+28 \* 2;

}

}

default\_cell\_fluids();

cells\_list.push(['pitocyte', spawnerz, spawnery, spawnerx, spawnerz + 6.18792735e+29, spawnery - 6.18792735e+29, spawnerx - 6.18792735e+29]);

}

}

function collagen() {

cursor4x = cursor1x;

cursor4y = cursor1y;

cursor4z = cursor1z;

peptidesequence = ' 1 mfsfvdlrll lllaatallt hgqeegqveg qdedippitc vqnglryhdr dvwkpepcri 61 cvcdngkvlc ddvicdetkn cpgaevpege ccpvcpdgse sptdqettgv egpkgdtgpr 121 gprgpagppg rdgipgqpgl pgppgppgpp gppglggnfa pqlsygydek stggisvpgp 181 mgpsgprglp gppgapgpqg fqgppgepge pgasgpmgpr gppgppgkng ddgeagkpgr 241 pgergppgpq garglpgtag lpgmkghrgf sgldgakgda gpagpkgepg spgengapgq 301 mgprglpger grpgapgpag argndgatga agppgptgpa gppgfpgavg akgeagpqgp 361 rgsegpqgvr gepgppgpag aagpagnpga dgqpgakgan gapgiagapg fpgargpsgp 421 qgpggppgpk gnsgepgapg skgdtgakge pgpvgvqgpp gpageegkrg argepgptgl 481 pgppgerggp gsrgfpgadg vagpkgpage rgspgpagpk gspgeagrpg eaglpgakgl 541 tgspgspgpd gktgppgpag qdgrpgppgp pgargqagvm gfpgpkgaag epgkagergv 601 pgppgavgpa gkdgeagaqg ppgpagpage rgeqgpagsp gfqglpgpag ppgeagkpge 661 qgvpgdlgap gpsgargerg fpgergvqgp pgpagprgan gapgndgakg dagapgapgs 721 qgapglqgmp gergaaglpg pkgdrgdagp kgadgspgkd gvrgltgpig ppgpagapgd 781 kgesgpsgpa gptgargapg drgepgppgp agfagppgad gqpgakgepg dagakgdagp 841 pgpagpagpp gpignvgapg akgargsagp pgatgfpgaa grvgppgpsg nagppgppgp 901 agkeggkgpr getgpagrpg evgppgppgp agekgspgad gpagapgtpg pqgiagqrgv 961 vglpgqrger gfpglpgpsg epgkqgpsga sgergppgpm gppglagppg esgregapga 1021 egspgrdgsp gakgdrgetg pagppgapga pgapgpvgpa gksgdrgetg pagpagpvgp 1081 vgargpagpq gprgdkgetg eqgdrgikgh rgfsglqgpp gppgspgeqg psgasgpagp 1141 rgppgsagap gkdglnglpg pigppgprgr tgdagpvgpp gppgppgppg ppsagfdfsf 1201 lpqppqekah dggryyradd anvvrdrdle vdttlkslsq qienirspeg srknpartcr 1261 dlkmchsdwk sgeywidpnq gcnldaikvf cnmetgetcv yptqpsvaqk nwyisknpkd 1321 krhvwfgesm tdgfqfeygg qgsdpadvai qltflrlmst easqnityhc knsvaymdqq 1381 tgnlkkalll qgsneieira egnsrftysv tvdgctshtg awgktvieyk ttktsrlpii 1441 dvapldvgap dqefgfdvgp vcfl';

protein();

cursor1x = cursor4x - 3.15584295e+25;

cursor1y = cursor4y - 2.475170941e+25;

cursor1z = cursor4z + 0;

peptidesequence = ' 1 mlsfvdtrtl lllavtlcla tcqslqeetv rkgpagdrgp rgergppgpp grdgedgptg 61 ppgppgppgp pglggnfaaq ydgkgvglgp gpmglmgprg ppgaagapgp qgfqgpagep 121 gepgqtgpag argpagppgk agedghpgkp grpgergvvg pqgargfpgt pglpgfkgir 181 ghngldglkg qpgapgvkge pgapgengtp gqtgarglpg ergrvgapgp agargsdgsv 241 gpvgpagpig sagppgfpga pgpkgeigav gnagpagpag prgevglpgl sgpvgppgnp 301 gangltgakg aaglpgvaga pglpgprgip gpvgaagatg arglvgepgp agskgesgnk 361 gepgsagpqg ppgpsgeegk rgpngeagsa gppgppglrg spgsrglpga dgragvmgpp 421 gsrgasgpag vrgpngdagr pgepglmgpr glpgspgnig pagkegpvgl pgidgrpgpi 481 gpagargepg nigfpgpkgp tgdpgkngdk ghaglagarg apgpdgnnga qgppgpqgvq 541 ggkgeqgppg ppgfqglpgp sgpagevgkp gerglhgefg lpgpagprge rgppgesgaa 601 gptgpigsrg psgppgpdgn kgepgvvgav gtagpsgpsg lpgergaagi pggkgekgep 661 glrgeignpg rdgargapga vgapgpagat gdrgeagaag pagpagprgs pgergevgpa 721 gpngfagpag aagqpgakge rgakgpkgen gvvgptgpvg aagpagpngp pgpagsrgdg 781 gppgmtgfpg aagrtgppgp sgisgppgpp gpagkeglrg prgdqgpvgr tgevgavgpp 841 gfagekgpsg eagtagppgt pgpqgllgap gilglpgsrg erglpgvaga vgepgplgia 901 gppgargppg avgspgvnga pgeagrdgnp gndgppgrdg qpghkgergy pgnigpvgaa 961 gapgphgpvg pagkhgnrge tgpsgpvgpa gavgprgpsg pqgirgdkge pgekgprglp 1021 glkghnglqg lpgiaghhgd qgapgsvgpa gprgpagpsg pagkdgrtgh pgtvgpagir 1081 gpqghqgpag ppgppgppgp pgvsgggydf gydgdfyrad qprsapslrp kdyevdatlk 1141 slnnqietll tpegsrknpa rtcrdlrlsh pewssgyywi dpnqgctmda ikvycdfstg 1201 etciraqpen ipaknwyrss kdkkhvwlge tinagsqfey nvegvtskem atqlafmrll 1261 anyasqnity hcknsiaymd eetgnlkkav ilqgsndvel vaegnsrfty tvlvdgcskk 1321 tnewgktiie yktnkpsrlp fldiapldig gadqeffvdi gpvcfk';

protein();

cursor1x = cursor4x - 3.15584295e+25;

cursor1y = cursor4y - 0;

cursor1z = cursor4z + 0;

protein();

peptidesequence = ' 1 mfsfvdlrll lllaatallt hgqeegqveg qdedippitc vqnglryhdr dvwkpepcri 61 cvcdngkvlc ddvicdetkn cpgaevpege ccpvcpdgse sptdqettgv egpkgdtgpr 121 gprgpagppg rdgipgqpgl pgppgppgpp gppglggnfa pqlsygydek stggisvpgp 181 mgpsgprglp gppgapgpqg fqgppgepge pgasgpmgpr gppgppgkng ddgeagkpgr 241 pgergppgpq garglpgtag lpgmkghrgf sgldgakgda gpagpkgepg spgengapgq 301 mgprglpger grpgapgpag argndgatga agppgptgpa gppgfpgavg akgeagpqgp 361 rgsegpqgvr gepgppgpag aagpagnpga dgqpgakgan gapgiagapg fpgargpsgp 421 qgpggppgpk gnsgepgapg skgdtgakge pgpvgvqgpp gpageegkrg argepgptgl 481 pgppgerggp gsrgfpgadg vagpkgpage rgspgpagpk gspgeagrpg eaglpgakgl 541 tgspgspgpd gktgppgpag qdgrpgppgp pgargqagvm gfpgpkgaag epgkagergv 601 pgppgavgpa gkdgeagaqg ppgpagpage rgeqgpagsp gfqglpgpag ppgeagkpge 661 qgvpgdlgap gpsgargerg fpgergvqgp pgpagprgan gapgndgakg dagapgapgs 721 qgapglqgmp gergaaglpg pkgdrgdagp kgadgspgkd gvrgltgpig ppgpagapgd 781 kgesgpsgpa gptgargapg drgepgppgp agfagppgad gqpgakgepg dagakgdagp 841 pgpagpagpp gpignvgapg akgargsagp pgatgfpgaa grvgppgpsg nagppgppgp 901 agkeggkgpr getgpagrpg evgppgppgp agekgspgad gpagapgtpg pqgiagqrgv 961 vglpgqrger gfpglpgpsg epgkqgpsga sgergppgpm gppglagppg esgregapga 1021 egspgrdgsp gakgdrgetg pagppgapga pgapgpvgpa gksgdrgetg pagpagpvgp 1081 vgargpagpq gprgdkgetg eqgdrgikgh rgfsglqgpp gppgspgeqg psgasgpagp 1141 rgppgsagap gkdglnglpg pigppgprgr tgdagpvgpp gppgppgppg ppsagfdfsf 1201 lpqppqekah dggryyradd anvvrdrdle vdttlkslsq qienirspeg srknpartcr 1261 dlkmchsdwk sgeywidpnq gcnldaikvf cnmetgetcv yptqpsvaqk nwyisknpkd 1321 krhvwfgesm tdgfqfeygg qgsdpadvai qltflrlmst easqnityhc knsvaymdqq 1381 tgnlkkalll qgsneieira egnsrftysv tvdgctshtg awgktvieyk ttktsrlpii 1441 dvapldvgap dqefgfdvgp vcfl';

cursor1x = cursor4x;

cursor1y = cursor4y;

cursor1z = cursor4z;

}

function mitochondrion() {

cursor4x = cursor1x;

cursor4y = cursor1y;

cursor4z = cursor1z;

if (unused\_variable == 5) {

cursor2x = cursor4x - 6.187927353e+26;

cursor2y = cursor4y - 6.187927353e+26;

cursor2z = cursor4z + 6.187927353e+26;

lipid1 = 2.970205129e+28;

lipid2 = 2.970205129e+28;

lipid\_wall\_z();

cursor2z = cursor4z + 5.940410259e+28;

lipid\_wall\_z();

cursor2z = cursor4z + 6.187927353e+26;

lipid2 = 5.878530986e+28;

lipid\_wall\_y();

lipid\_wall\_x();

cursor2y = cursor4y - 3.032084403e+28;

lipid\_wall\_y();

cursor2x = cursor4x - 3.032084403e+28;

cursor2y = cursor4y - 6.187927353e+26;

lipid\_wall\_x();

cursor2x = cursor4x - 6.187927353e+26;

}

if (unused\_variable == 5) {

cursor2x = cursor4x - 1.23758547e+28;

cursor2y = cursor4y - 1.856378206e+27;

cursor2z = cursor4z + 1.23758547e+27;

lipid1 = 6.187927353e+27;

lipid2 = 2.722688035e+28;

lipid\_wall\_z();

cursor2z = cursor4z + 5.816651712e+28;

lipid\_wall\_z();

cursor2z = cursor4z + 1.23758547e+27;

lipid2 = 5.692893165e+28;

lipid\_wall\_y();

cursor2y = cursor4y - 2.846446582e+28;

lipid\_wall\_y();

}

if (unused\_variable == 5) {

cursor2x = cursor4x - 1.856378206e+27;

cursor2y = cursor4y - 1.856378206e+27;

cursor2z = cursor4z + 1.23758547e+27;

for (var count = 0; count < 23; count++) {

lipid1 = 1.05194765e+28;

lipid2 = 2.722688035e+28;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+27;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+27;

lipid1 = 1.05194765e+28;

lipid2 = 1.23758547e+27;

lipid\_wall\_y();

cursor2y = cursor2y - 2.722688035e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 2.722688035e+28;

lipid1 = 2.722688035e+28;

lipid\_wall\_x();

cursor2z = cursor2z + 1.23758547e+27;

cursor2x = cursor2x - 1.05194765e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.05194765e+28;

cursor2z = cursor2z + 1.23758547e+27;

}

cursor2x = cursor4x - 1.856378206e+28;

cursor2y = cursor4y - 1.856378206e+27;

cursor2z = cursor4z + 1.23758547e+27;

for (var count2 = 0; count2 < 23; count2++) {

lipid1 = 1.05194765e+28;

lipid2 = 2.722688035e+28;

lipid\_wall\_z();

cursor2z = cursor2z + 1.23758547e+27;

lipid\_wall\_z();

cursor2z = cursor2z - 1.23758547e+27;

lipid1 = 1.05194765e+28;

lipid2 = 1.23758547e+27;

lipid\_wall\_y();

cursor2y = cursor2y - 2.722688035e+28;

lipid\_wall\_y();

cursor2y = cursor2y + 2.722688035e+28;

lipid1 = 2.722688035e+28;

cursor2x = cursor2x - 1.05194765e+28;

lipid\_wall\_x();

cursor2x = cursor2x + 1.05194765e+28;

cursor2z = cursor2z + 1.23758547e+27;

lipid\_wall\_x();

cursor2z = cursor2z + 1.23758547e+27;

}

}

if (unused\_variable == 5) {

cursor2x = cursor4x - 1.299464744e+28;

cursor2y = cursor4y - 1.23758547e+28;

cursor2z = cursor4z + 1.856378206e+27;

genomeprevious = genome;

genome = mitochondrial\_genome;

ploidyprevious = ploidy;

ploidy = 1;

nucleoid\_size\_y = 6.187927353e+27;

nucleoid\_size\_z = 5.569134618e+28;

nucleoid();

genome = genomeprevious;

ploidy = ploidyprevious;

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 1.23758547e+28;

cursor1y = cursor4y - 1.918257479e+28;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count5 = 0; count5 < generation\_parameter\_3; count5++) {

for (var count4 = 0; count4 < generation\_parameter\_x; count4++) {

for (var count3 = 0; count3 < generation\_parameter\_z; count3++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

coenzyme\_A();

} else if (randomizer <= 20) {

coenzyme\_Q();

} else if (randomizer <= 25) {

lipoic\_acid();

} else if (randomizer <= 30) {

TPP();

} else {

randomizer = math\_random\_int(1, 80);

if (randomizer <= 10) {

randomizer = math\_random\_int(1, 105);

if (randomizer <= 15) {

peptidesequence = ' 1 mrkmlaavsr vlsgasqkpa srvlvasrnf andatfeikk cdlhrleegp pvttvltred 61 glkyyrmmqt vrrmelkadq lykqkiirgf chlcdgqeac cvgleaginp tdhlitayra 121 hgftftrgls vreilaeltg rkggcakgkg gsmhmyaknf yggngivgaq vplgagiala 181 ckyngkdevc ltlygdgaan qgqifeaynm aalwklpcif icennrygmg tsveraaast 241 dyykrgdfip glrvdgmdil cvreatrfaa aycrsgkgpi lmelqtyryh ghsmsdpgvs 301 yrtreeiqev rsksdpimll kdrmvnsnla sveelkeidv evrkeiedaa qfatadpepp 361 leelgyhiys sdppfevrga nqwikfksvs';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mlaafisrvl rrvaqksarr vlvasrnssn datfeikkcd lylleegppv ttvltraegl 61 kyyrmmltvr rmelkadqly kqkfirgfch lcdgqeaccv gleaginpsd hvitsyrahg 121 vcytrglsvr silaeltgrr ggcakgkggs mhmytknfyg gngivgaqgp lgagialack 181 ykgndeiclt lygdgaanqg qiaeafnmaa lwklpcvfic ennlygmgts teraaaspdy 241 ykrgnfipgl kvdgmdvlcv reatkfaany crsgkgpilm elqtyryhgh smsdpgvsyr 301 treeiqevrs krdpiiilqd rmvnsklatv eelkeigaev rkeiddaaqf attdpephle 361 elghhiyssd ssfevrganp wikfksvs';

protein();

} else if (randomizer <= 45) {

peptidesequence = ' 1 maavsglvrr plrevsgllk rrfhwtapaa lqvtvrdain qgmdeelerd ekvfllgeev 61 aqydgaykvs rglwkkygdk riidtpisem gfagiavgaa maglrpicef mtfnfsmqai 121 dqvinsaakt yymsgglqpv pivfrgpnga sagvaaqhsq cfaawyghcp glkvvspwns 181 edakgliksa irdnnpvvvl enelmygvpf efppeaqskd flipigkaki erqgthitvv 241 shsrpvghcl eaaavlskeg vecevinmrt irpmdmetie asvmktnhlv tveggwpqfg 301 vgaeicarim egpafnflda pavrvtgadv pmpyakiled nsipqvkdii faikktlni';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 serfpndvdp ietrdwlqai esvireegve raqylidqll aearkggvnv aagtgisnyi 61 ntipveeqpe ypgnlelerr irsairwnai mtvlraskkd lelgghmasf qssatiydvc 121 fnhffrarne qdggdlvyfq ghispgvyar aflegrltqe qldnfrqevh gnglssyphp 181 klmpefwqfp tvsmglgpig aiyqakflky lehrglkdts kqtvyaflgd gemdepeskg 241 aitiatrekl dnlvfvincn lqrldgpvtg ngkiineleg ifegagwnvi kvmwgsrwde 301 llrkdtsgkl iqlmnetvdg dyqtfkskdg ayvrehffgk ypetaalvad wtdeqiwaln 361 rgghdpkkiy aafkkaqetk gkatvilaht ikgygmgdaa egkniahqvk kmnmdgvrhi 421 rdrfnvpvsd adieklpyit fpegseehty lhaqrqklhg ylpsrqpnft eklelpslqd 481 fgalleeqsk eisttiafvr alnvmlknks ikdrlvpiia deartfgmeg lfrqigiysp 541 ngqqytpqdr eqvayykede kgqilqegin elgagcswla aatsystnnl pmipfyiyys 601 mfgfqrigdl cwaagdqqar gfliggtsgr ttlngeglqh edghshiqsl tipncisydp 661 ayayevavim hdglermyge kqenvyyyit tlnenyhmpa mpegaeegir kgiykletie 721 gskgkvqllg sgsilrhvre aaeilakdyg vgsdvysvts ftelardgqd cerwnmlhpl 781 etprvpyiaq vmndapavas tdymklfaeq vrtyvpaddy rvlgtdgfgr sdsrenlrhh 841 fevdasyvvv aalgelakrg eidkkvvada iakfnidadk vnprla';

protein();

} else if (randomizer <= 75) {

peptidesequence = ' 1 mwrvcarraq nvapwaglea rwtalqevpg tprvtsrsgp aparrnsvtt gyggvralcg 61 wtpssgatpr nrlllqllgs pgrryyslpp hqkvplpsls ptmqagtiar wekkegdkin 121 egdliaevet dkatvgfesl eecymakilv aegtrdvpig aiicitvgkp edieafknyt 181 ldssaaptpq aapaptpaat aspptpsaqa pgssypphmq vllpalsptm tmgtvqrwek 241 kvgeklsegd llaeietdka tigfevqeeg ylakilvpeg trdvplgtpl ciivekeadi 301 safadyrpte vtdlkpqvpp ptpppvaavp ptpqplaptp sapcpatpag pkgrvfvspl 361 akklavekgi dltqvkgtgp dgritkkdid sfvpskvapa paavvpptgp gmapvptgvf 421 tdipisnirr viaqrlmqsk qtiphyylsi dvnmgevllv rkelnkffls fqilegrski 481 svndfiikas alaclkvpea nsswmdtvir qnhvvdvsva vstpaglitp ivfnahikgv 541 etiandvvsl atkaregklq phefqggtft isnlgmfgik nfsaiinppq acilaigase 601 dklvpadnek gfdvasmmsv tlscdhrvvd gavgaqwlae frkylekpit mll';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mahgkdfasr giemsevrln ldkmmeqkst avkaltggia hlfkqnkvvh vngygkitgk 61 nqvtatkadg gtqvidtkni liatgsevtp fpgitidedt ivsstgalsl kkvpekmvvi 121 gagvigvelg svwqrlgadv taveflghvg gvgidmeisk nfqrilqkqg fkfklntkvt 181 gatkksdgki dvsieaasgg kaevitcdvl lvcigrrpft knlgleelgi eldprgripv 241 ntrfqtkipn iyaigdvvag pmlahkaede giicvegmag gavhidyncv psviythpev 301 awvgkseeql keegieykvg kfpfaansra ktnadtdgmv kilgqkstdr vlgahilgpg 361 agemvneaal aleygasced iarvchahpt lseafreanl aasfgksinf';

protein();

} else {

peptidesequence = ' 1 mqswsrvycs lakrghfnri shglqglsav plrtyadqpi dadvtvigsg pggyvaaika 61 aqlgfktvci eknetlggtc lnvgcipska llnnshyyhm ahgkdfasrg iemsevrlnl 121 dkmmeqksta vkaltggiah lfkqnkvvhv ngygkitgkn qvtatkadgg tqvidtknil 181 iatgsevtpf pgitidedti vsstgalslk kvpekmvvig agvigvelgs vwqrlgadvt 241 aveflghvgg vgidmeiskn fqrilqkqgf kfklntkvtg atkksdgkid vsieaasggk 301 aevitcdvll vcigrrpftk nlgleelgie ldprgripvn trfqtkipni yaigdvvagp 361 mlahkaedeg iicvegmagg avhidyncvp sviythpeva wvgkseeqlk eegieykvgk 421 fpfaansrak tnadtdgmvk ilgqkstdrv lgahilgpga gemvneaala leygascedi 481 arvchahptl seafreanla asfgksinf';

protein();

}

} else if (randomizer <= 20) {

peptidesequence = ' 1 malltaaarl lgtknasclv laarhasass tnlkdiladl ipkeqarikt frqqhgktvv 61 gqitvdmmyg gmrgmkglvy etsvldpdeg irfrgfsipe cqkllpkakg geeplpeglf 121 wllvtghipt eeqvswlske wakraalpsh vvtmldnfpt nlhpmsqlsa avtalnsesn 181 farayaqgis rtkyweliye dsmdliaklp cvaakiyrnl yregsgigai dsnldwshnf 241 tnmlgytdhq fteltrlylt ihsdheggnv sahtshlvgs alsdpylsfa aamnglagpl 301 hglanqevlv wltqlqkevg kdvsdeklrd yiwntlnsgr vvpgyghavl rktdprytcq 361 refalkhlpn dpmfklvaql ykivpnvlle qgkaknpwpn vdahsgvllq yygmtemnyy 421 tvlfgvsral gvlaqliwsr algfplerpk smsteglmkf vdsksg';

protein();

} else if (randomizer <= 30) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 15) {

iron\_sulphur\_cluster();

} else if (randomizer <= 30) {

SO4\_anion();

} else if (randomizer <= 45) {

peptidesequence = ' 1 mshfepneyi hydlleknin ivrkrlnrpl tlsekivygh lddpasqeie rgksylrlrp 61 drvamqdata qmamlqfiss glskvavpst ihcdhlieaq vggekdlrra kdinqevynf 121 latagakygv gfwkpgsgii hqiilenyay pgvlligtds htpnggglgg icigvggada 181 vdvmagipwe lkcpkvigvk ltgslsgwss pkdvilkvag iltvkggtga iveyhgpgvd 241 sisctgmati cnmgaeigat tsvfpynhrm kkylsktgre dianladefk dhlvpdpgch 301 ydqlieinls elkphingpf tpdlahpvae vgkvaekegw pldirvglig sctnssyedm 361 grsaavakqa lahglkcksq ftitpgseqi ratierdgya qilrdlggiv lanacgpcig 421 qwdrkdikkg ekntivtsyn rnftgrndan pethafvtsp eivtalaiag tlkfnpetdy 481 ltgtdgkkfr leapdadelp kgefdpgqdt yqhppkdssg qhvdvsptsq rlqllepfdk 541 wdgkdledlq ilikvkgkct tdhisaagpw lkfrghldni snnlligain iengkansvr 601 navtqefgpv pdtaryykkh girwvvigde nygegssreh aaleprhlgg raiitksfar 661 ihetnlkkqg llpltfadpa dynkihpvdk ltiqglkdft pgkplkciik hpngtqetil 721 lnhtfnetqi ewfragsaln rmkelqq';

protein();

} else {

peptidesequence = ' 1 mapysllvtr lqkalgvrqy hvasvlcqra kvamshfepn eyihydllek ninivrkrln 61 rpltlsekiv yghlddpasq eiergksylr lrpdrvamqd ataqmamlqf issglskvav 121 pstihcdhli eaqvggekdl rrakdinqev ynflatagak ygvgfwkpgs giihqiilen 181 yaypgvllig tdshtpnggg lggicigvgg adavdvmagi pwelkcpkvi gvkltgslsg 241 wsspkdvilk vagiltvkgg tgaiveyhgp gvdsisctgm aticnmgaei gattsvfpyn 301 hrmkkylskt gredianlad efkdhlvpdp gchydqliei nlselkphin gpftpdlahp 361 vaevgkvaek egwpldirvg ligsctnssy edmgrsaava kqalahglkc ksqftitpgs 421 eqiratierd gyaqilrdlg givlanacgp cigqwdrkdi kkgekntivt synrnftgrn 481 danpethafv tspeivtala iagtlkfnpe tdyltgtdgk kfrleapdad elpkgefdpg 541 qdtyqhppkd ssgqhvdvsp tsqrlqllep fdkwdgkdle dlqilikvkg kcttdhisaa 601 gpwlkfrghl dnisnnllig ainiengkan svrnavtqef gpvpdtaryy kkhgirwvvi 661 gdenygegss rehaaleprh lggraiitks farihetnlk kqgllpltfa dpadynkihp 721 vdkltiqglk dftpgkplkc iikhpngtqe tillnhtfne tqiewfrags alnrmkelqq';

protein();

}

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 10) {

peptidesequence = ' 1 magpawiskv srllgafhnp kqvtrgftgg vqtvtlipgd gigpeisaav mkifdaakap 61 iqweernvta iqgpggkwmi pseakesmdk nkmglkgplk tpiaaghpsm nlllrktfdl 121 yanvrpcvsi egyktpytdv nivtirente geysgiehvi vdgvvqsikl itegaskria 181 efafeyarnn hrsnvtavhk animrmsdgl flqkcrevae sckdikfnem yldtvclnmv 241 qdpsqfdvlv mpnlygdils dlcagliggl gvtpsgniga ngvaifesvh gtapdiagkd 301 manptallls avmmlrhmgl fdhaarieaa cfatikdgks ltkdlggnak csdfteeicr 361 rvkdld';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 mkifdaakap iqweernvta iqgpggkwmi pseakesmdk nkmglkgplk tpiaaghpsm 61 nlllrktfdl yanvrpcvsi egyktpytdv nivtirente geysgiehvi vdgvvqsikl 121 itegaskria efafeyarnn hrsnvtavhk animrmsdgl flqkcrevae sckdikfnem 181 yldtvclnmv qdpsqfdvlv mpnlygdils dlcagliggl gvtpsgniga ngvaifesvh 241 gtapdiagkd manptallls avmmlrhmgl fdhaarieaa cfatikdgks ltkdlggnak 301 csdfteeicr rvkdld';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 maalsgvrwl tralvsagnp gawrglstsa aahaasrsqa edvrvegsfp vtmlpgdgvg 61 pelmhavkev fkaaavpvef qehhlsevqn maseekleqv lssmkenkva iigkihtpme 121 ykgelasydm rlrrkldlfa nvvhvkslpg ymtrhnnldl viireqtege ysslehesar 181 gvieclkivt raksqriakf afdyatkkgr gkvtavhkan imklgdglfl qcceevaely 241 pkikfetmii dnccmqlvqn pyqfdvlvmp nlygniidnl aaglvggagv vpgesysaey 301 avfetgarhp faqavgrnia nptamllsas nmlrhlnley hssmiadavk kvikvgkvrt 361 rdmggysttt dfiksvighl qtkgs';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 maalsgvrwl tralvsagnp gawrglstsa aahaasrsqa edvrvegsfp vtmlpgdgvg 61 pelmhavkev fkaaavpvef qehhlsevqn maseekleqv lssmkenkva iigkihtpme 121 ykgelasydm rlrrkldlfa nvvhvkslpg ymtrhnnldl viireqtege ysslehesar 181 gvieclkivt raksqriakf afdyatkkgr gkvtavhkan imklgdglfl qcceevaely 241 pkikfetmii dnccmqlvqn pyqfdvlvmp nlygniidnl aaglvggagv vpgesysaey 301 avfetgarhp faqavgrnia nptamllsas nmlrhlnley hssmiadavk kvikvgkvrt 361 rdmggysttt dfiksvighl qtkgs';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 malkvatvag saakavlgpa llcrpwevlg ahevpsrnif seqtippsak yggrhtvtmi 61 pgdgigpelm lhvksvfrha cvpvdfeevh vssnadeedi rnaimairrn rvalkgniet 121 nhnlppshks rnnilrtsld lyanvihcks lpgvvtrhkd idilivrent egeysslehe 181 svagvveslk iitkakslri aeyafklaqe sgrkkvtavh kanimklgdg lflqccreva 241 arypqitfen mivdnttmql vsrpqqfdvm vmpnlygniv nnvcaglvgg pglvaganyg 301 hvyavfetat rntgksiank nianptatll ascmmldhlk lhsyatsirk avlasmdnen 361 mhtpdiggqg ttseaiqdvi rhirvingra vea';

protein();

} else {

peptidesequence = ' 1 malkvatvag saakavlgpa llcrpwevlg ahevpsrnif seqtippsak yggrhtvtmi 61 pgdgigpelm lhvksvfrha cvpvdfeevh vssnadeedi rnaimairrn rvalkgniet 121 nhnlppshks rnnilrtsld lyanvihcks lpgvvtrhkd idilivrent egeysslehe 181 svagvveslk iitkakslri aeyafklaqe sgrkkvtavh kanimklgdg lflqccreva 241 arypqitfen mivdnttmql vsrpqqfdvm vmpnlygniv nnvcaglvgg pglvaganyg 301 hvyavfetat rntgksiank nianptatll ascmmldhlk lhsyatsirk avlasmdnen 361 mhtpdiggqg ttseaiqdvi rhirvingra vea';

protein();

}

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 10) {

peptidesequence = ' 1 mfhlrtcaak lrpltasqtv ktfsqnrpaa artfqqircy sapvaaepfl sgtssnyvee 61 mycawlenpk svhkswdiff rntnagappg tayqsplpls rgslaavaha qslveaqpnv 121 dklvedhlav qslirayqir ghhvaqldpl gildadldss vpadiisstd klgfygldes 181 dldkvfhlpt ttfiggqesa lplreiirrl emaycqhigv efmfindleq cqwirqkfet 241 pgimqftnee krtllarlvr strfeeflqr kwssekrfgl egcevlipal ktiidkssen 301 gvdyvimgmp hrgrlnvlan virkeleqif cqfdskleaa degsgdvkyh lgmyhrrinr 361 vtdrnitlsl vanpshleaa dpvvmgktka eqfycgdteg kkvmsillhg daafagqgiv 421 yetfhlsdlp sytthgtvhv vvnnqigftt dprmarsspy ptdvarvvna pifhvnsddp 481 eavmyvckva aewrstfhkd vvvdlvcyrr nghnemdepm ftqplmykqi rkqkpvlqky 541 aellvsqgvv nqpeyeeeis kydkiceeaf arskdekilh ikhwldspwp gfftldgqpr 601 smscpstglt edilthignv assvpvenft ihgglsrilk trgemvknrt vdwalaeyma 661 fgsllkegih irlsgqdver gtfshrhhvl hdqnvdkrtc ipmnhlwpnq apytvcnssl 721 seygvlgfel gfamaspnal vlweaqfgdf hntaqciidq ficpgqakwv rqngivlllp 781 hgmegmgpeh ssarperflq mcnddpdvlp dlkeanfdin qlydcnwvvv ncstpgnffh 841 vlrrqillpf rkpliiftpk sllrhpears sfdemlpgth fqrvipedgp aaqnpenvkr 901 llfctgkvyy dltrerkard mvgqvaitri eqlspfpfdl llkevqkypn aelawcqeeh 961 knqgyydyvk prlrttisra kpvwyagrdp aaapatgnkk thltelqrll dtafdldvfk 1021 nfs';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 mfhlrtcaak lrpltasqtv ktfsqnrpaa artfqqircy sapvaaepfl sgtssnyvee 61 mycawlenpk svhkswdiff rntnagappg tayqsplpls rgslaavaha qslveaqpnv 121 dklvedhlav qslirayqir ghhvaqldpl gildadldss vpadiisstd kldlavfker 181 lrmltvggfy gldesdldkv fhlptttfig gqesalplre iirrlemayc qhigvefmfi 241 ndleqcqwir qkfetpgimq ftneekrtll arlvrstrfe eflqrkwsse krfglegcev 301 lipalktiid kssengvdyv imgmphrgrl nvlanvirke leqifcqfds kleaadegsg 361 dvkyhlgmyh rrinrvtdrn itlslvanps hleaadpvvm gktkaeqfyc gdtegkkvms 421 illhgdaafa gqgivyetfh lsdlpsytth gtvhvvvnnq igfttdprma rsspyptdva 481 rvvnapifhv nsddpeavmy vckvaaewrs tfhkdvvvdl vcyrrnghne mdepmftqpl 541 mykqirkqkp vlqkyaellv sqgvvnqpey eeeiskydki ceeafarskd ekilhikhwl 601 dspwpgfftl dgqprsmscp stgltedilt hignvassvp venftihggl srilktrgem 661 vknrtvdwal aeymafgsll kegihirlsg qdvergtfsh rhhvlhdqnv dkrtcipmnh 721 lwpnqapytv cnsslseygv lgfelgfama spnalvlwea qfgdfhntaq ciidqficpg 781 qakwvrqngi vlllphgmeg mgpehssarp erflqmcndd pdvlpdlkea nfdinqlydc 841 nwvvvncstp gnffhvlrrq illpfrkpli iftpksllrh pearssfdem lpgthfqrvi 901 pedgpaaqnp envkrllfct gkvyydltre rkardmvgqv aitrieqlsp fpfdlllkev 961 qkypnaelaw cqeehknqgy ydyvkprlrt tisrakpvwy agrdpaaapa tgnkkthlte 1021 lqrlldtafd ldvfknfs';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mlsrsrcvsr afsrslsafq kgncplgrrs lpgvslcqgp gypnsrkvvi nnsvfsvrff 61 rttavckddl vtvktpafae svtegdvrwe kavgdtvaed evvceietdk tsvqvpspan 121 gvieallvpd ggkveggtpl ftlrktgaap akakpaeapa aaapkaepta aavpppaapi 181 ptqmppvpsp sqppsgkpvs avkptvappl aepgagkglr sehrekmnrm rqriaqrlke 241 aqntcamltt fneidmsniq emrarhkeaf lkkhnlklgf msafvkasaf alqeqpvvna 301 viddttkevv yrdyidisva vatprglvvp virnveamnf adiertitel gekarknela 361 iedmdggtft isnggvfgsl fgtpiinppq sailgmhgif drpvaiggkv evrpmmyval 421 tydhrlidgr eavtflrkik aavedprvll ldl';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mlsrsrcvsr afsrslsafq kgncplgrrs lpgvslcqgp gypnsrkvvi nnsvfsvrff 61 rttavckddl vtvktpafae svtegdvrwe kavgdtvaed evvceietdk tsvqvpspan 121 gvieallvpd ggkveggtpl ftlrktgaap akakpaeapa aaapkaepta aavpppaapi 181 ptqmppvpsp sqppsgkpvs avkptvappl aepgagkglr sehrekmnrm rqriaqrlke 241 aqntcamltt fneidmsniq emrarhkeaf lkkhnlklgf msafvkasaf alqeqpvvna 301 viddttkevv yrdyidisva vatprglvvp virnveamnf adiertitel gekarknela 361 iedmdggtft isnggvfgsl fgtpiinppq sailgmhgif drpvaiggkv evrpmmyval 421 tydhrlidgr eavtflrkik aavedprvll ldl';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 mqswsrvycs lakrghfnri shglqglsav plrtyadqpi dadvtvigsg pggyvaaika 61 aqlgfktvci eknetlggtc lnvgcipska llnnshyyhm ahgkdfasrg iemsevrlnl 121 dkmmeqksta vkaltggiah lfkqnkvvhv ngygkitgkn qvtatkadgg tqvidtknil 181 iatgsevtpf pgitidedti vsstgalslk kvpekmvvig agvigvelgs vwqrlgadvt 241 aveflghvgg vgidmeiskn fqrilqkqgf kfklntkvtg atkksdgkid vsieaasggk 301 aevitcdvll vcigrrpftk nlgleelgie ldprgripvn trfqtkipni yaigdvvagp 361 mlahkaedeg iicvegmagg avhidyncvp sviythpeva wvgkseeqlk eegieykvgk 421 fpfaansrak tnadtdgmvk ilgqkstdrv lgahilgpga gemvneaala leygascedi 481 arvchahptl seafreanla asfgksinf';

protein();

} else {

peptidesequence = ' 1 mqswsrvycs lakrghfnri shglqglsav plrtyadqpi dadvtvigsg pggyvaaika 61 aqlgfkalln nshyyhmahg kdfasrgiem sevrlnldkm meqkstavka ltggiahlfk 121 qnkvvhvngy gkitgknqvt atkadggtqv idtkniliat gsevtpfpgi tidedtivss 181 tgalslkkvp ekmvvigagv igvelgsvwq rlgadvtave flghvggvgi dmeisknfqr 241 ilqkqgfkfk lntkvtgatk ksdgkidvsi eaasggkaev itcdvllvci grrpftknlg 301 leelgieldp rgripvntrf qtkipniyai gdvvagpmla hkaedegiic vegmaggavh 361 idyncvpsvi ythpevawvg kseeqlkeeg ieykvgkfpf aansraktna dtdgmvkilg 421 qkstdrvlga hilgpgagem vneaalaley gascediarv chahptlsea freanlaasf 481 gksinf';

protein();

}

} else if (randomizer <= 60) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 10) {

peptidesequence = '1 mtatlaaaad iatmvsgssg laaarllsrs fllpqngirh csytasrqhl yvdkntkiic 61 qgftgkqgtf hsqqaleygt klvggttpgk ggqthlglpv fntvkeakeq tgatasviyv 121 pppfaaaain eaieaeiplv vcitegipqq dmvrvkhkll rqektrligp ncpgvinpge 181 ckigimpghi hkkgrigivs rsgtltyeav hqttqvglgq slcvgiggdp fngtdfidcl 241 eiflndsate giiligeigg naeenaaefl kqhnsgpnsk pvvsfiaglt appgrrmgha 301 gaiiaggkgg akekisalqs agvvvsmspa qlgttiykef ekrkml';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mtatlaaaad iatmvsgssg laaarllsrs fllpqngirh csytasrqhl yvdkntkiic 61 qgftgkqgtf hsqqaleygt klvggttpgk ggqthlglpv fntvkeakeq tgatasviyv 121 pppfaaaain eaieaeiplv vcitegipqq dmvrvkhkll rqektrligp ncpgvinpge 181 ckigimpghi hkkgrigivs rsgtltyeav hqttqvglgq slcvgiggdp fngtdfidcl 241 eiflndsate giiligeigg naeenaaefl kqhnsgpnsk pvvsfiaglt appgrrmgha 301 gaiiaggkgg akekisalqs agvvvsmspa qlgttiykef ekrkml';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mtatlaaaad iatmvsgssg laaarllsrs fllpqngirh csytasrqhl yvdkntkiic 61 qgftgkqgtf hsqqaleygt klvggttpgk ggqthlglpv fntvkeakeq tgatasviyv 121 pppfaaaain eaieaeiplv vcitegipqq dmvrvkhkll rqektrligp ncpgvinpge 181 ckigimpghi hkkgrigivs rsgtltyeav hqttqvglgq slcvgiggdp fngtdfidcl 241 eiflndsate giiligeigg naeenaaefl kqhnsgpnsk pvvsfiaglt appgrrmgha 301 gaiiaggkgg akekisalqs agvvvsmspa qlgttiykef ekrkml';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 maasmfygrl vavatlrnhr prtaqraaaq vlgssglfnn hglqvqqqqq rnlslheyms 61 mellqeagvs vpkgyvaksp deayaiakkl gskdvvikaq vlaggrgkgt fesglkggvk 121 ivfspeeaka vssqmigkkl ftkqtgekgr icnqvlvcer kyprreyyfa itmersfqgp 181 vligsshggv niedvaaesp eaiikepidi eegikkeqal qlaqkmgfpp nivesaaenm 241 vklyslflky datmieinpm vedsdgavlc mdakinfdsn sayrqkkifd lqdwtqeder 301 dkdaakanln yigldgnigc lvngaglama tmdiiklhgg tpanfldvgg gatvhqvtea 361 fklitsdkkv lailvnifgg imrcdviaqg ivmavkdlei kipvvvrlqg trvddakali 421 adsglkilac ddldeaarmv vklseivtla kqahvdvkfq lpi';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 maasmfygrl vavatlrnhr prtaqraaaq vlgssglfnn hglqvqqqqq rnlslheyms 61 mellqeagvs vpkgyvaksp deayaiakkl gskdvvikaq vlaggrgkgt fesglkggvk 121 ivfspeeaka vssqmigkkl ftkqtgekgr icnqvlvcer kyprreyyfa itmersfqgp 181 vligsshggv niedvaaesp eaiikepidi eegikkeqal qlaqkmgfpp nivesaaenm 241 vklyslflky datmieinpm vedsdgavlc mdakinfdsn sayrqkkifd lqdwtqeder 301 dkdaakanln yigldgnigc lvngaglama tmdiiklhgg tpanfldvgg gatvhqvtea 361 fklitsdkkv lailvnifgg imrcdviaqg ivmavkdlei kipvvvrlqg trvddakali 421 adsglkilac ddldeaarmv vklseivtla kqahvdvkfq lpi';

protein();

} else {

peptidesequence = '1 maasmfygrl vavatlrnhr prtaqraaaq vlgssglfnn hglqvqqqqq rnlslheyms 61 mellqeagvs vpkgyvaksp deayaiakkl gskdvvikaq vlaggrgkgt fesglkggvk 121 ivfspeeaka vssqmigkkl ftkqtgekgr icnqvlvcer kyprreyyfa itmersfqgp 181 vligsshggv niedvaaesp eaiikepidi eegikkeqal qlaqkmgfpp nivesaaenm 241 vklyslflky datmieinpm vedsdgavlc mdakinfdsn sayrqkkifd lqdwtqeder 301 dkdaakanln yigldgnigc lvngaglama tmdiiklhgg tpanfldvgg gatvhqvtea 361 fklitsdkkv lailvnifgg imrcdviaqg ivmavkdlei kipvvvrlqg trvddakali 421 adsglkilac ddldeaarmv vklseivtla kqahvdvkfq lpi';

protein();

}

} else if (randomizer <= 70) {

peptidesequence = ' 1 myralrllar srplvrapaa alasapglgg aavpsfwppn aarmasqnsf rieydtfgel 61 kvpndkyyga qtvrstmnfk iggvtermpt pvikafgilk raaaevnqdy gldpkianai 121 mkaadevaeg klndhfplvv wqtgsgtqtn mnvnevisnr aiemlggelg skipvhpndh 181 vnksqssndt fptamhiaaa ievhevllpg lqklhdalda kskefaqiik igrthtqdav 241 pltlgqefsg yvqqvkyamt rikaampriy elaaggtavg tglntrigfa ekvaakvaal 301 tglpfvtapn kfealaahda lvelsgamnt tacslmkian dirflgsgpr sglgelilpe 361 nepgssimpg kvnptqceam tmvaaqvmgn hvavtvggsn ghfelnvfkp mmiknvlhsa 421 rllgdasvsf tencvvgiqa nterinklmn eslmlvtaln phigydkaak iaktahkngs 481 tlketaielg yltaeqfdew vkpkdmlgpk';

protein();

} else {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 10) {

peptidesequence = '1 mlsalarpas aalrrsfsts aqnnakvavl gasggigqpl slllknsplv srltlydiah 61 tpgvaadlsh ietkaavkgy lgpeqlpdcl kgcdvvvipa gvprkpgmtr ddlfntnati 121 vatltaacaq hcpeamicvi anpvnstipi taevfkkhgv ynpnkifgvt tldivrantf 181 vaelkgldpa rvnvpviggh agktiiplis qctpkvdfpq dqltaltgri qeagtevvka 241 kagagsatls mayagarfvf slvdamngke gvvecsfvks qetectyfst plllgkkgie 301 knlgigkvss feekmisdai pelkasikkg edfvktlk';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mlsalarpas aalrrsfsts aqnnakvavl gasggigqpl slllknsplv srltlydiah 61 tpgvaadlsh ietkaavkgy lgpeqlpdcl kgcdvvvipa gvprkpgmtr ddlfntnati 121 vatltaacaq hcpeamicvi anpvnstipi taevfkkhgv ynpnkifgvt tldivrantf 181 vaelkgldpa rvnvpviggh agktiiplis qctpkvdfpq dqltaltgri qeagtevvka 241 kagagsatls mayagarfvf slvdamngke gvvecsfvks qetectyfst plllgkkgie 301 knlgigkvss feekmisdai pelkasikkg edfvktlk';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mtrddlfntn ativatltaa caqhcpeami cvianpvnst ipitaevfkk hgvynpnkif 61 gvttldivra ntfvaelkgl dparvnvpvi gghagktiip lisqctpkvd fpqdqltalt 121 griqeagtev vkakagagsa tlsmayagar fvfslvdamn gkegvvecsf vksqetecty 181 fstplllgkk gieknlgigk vssfeekmis daipelkasi kkgedfvktl k';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mtrddlfntn ativatltaa caqhcpeami cvianpvnst ipitaevfkk hgvynpnkif 61 gvttldivra ntfvaelkgl dparvnvpvi gghagktiip lisqctpkvd fpqdqltalt 121 griqeagtev vkakagagsa tlsmayagar fvfslvdamn gkegvvecsf vksqetecty 181 fstplllgkk gieknlgigk vssfeekmis daipelkasi kkgedfvktl k';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mlsalarpas aalrrsfsts aqnnakvavl gasggigqpl slllknsplv srltlydiah 61 tpgvaadlsh ietkaavkgy lgpeqlpdcl kgcdvvvipa gvprkpgmtr ddlfntnati 121 vatltaacaq hcpeamicvi anpgldparv nvpvigghag ktiiplisqc tpkvdfpqdq 181 ltaltgriqe agtevvkaka gagsatlsma yagarfvfsl vdamngkegv vecsfvksqe 241 tectyfstpl llgkkgiekn lgigkvssfe ekmisdaipe lkasikkged fvktlk';

protein();

} else {

peptidesequence = ' 1 mlsalarpas aalrrsfsts aqnnakvavl gasggigqpl slllknsplv srltlydiah 61 tpgvaadlsh ietkaavkgy lgpeqlpdcl kgcdvvvipa gvprkpgmtr ddlfntnati 121 vatltaacaq hcpeamicvi anpgldparv nvpvigghag ktiiplisqc tpkvdfpqdq 181 ltaltgriqe agtevvkaka gagsatlsma yagarfvfsl vdamngkegv vecsfvksqe 241 tectyfstpl llgkkgiekn lgigkvssfe ekmisdaipe lkasikkged fvktlk';

protein();

}

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 1.23758547e+28;

cursor1y = cursor4y - 1.856378206e+27;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(6.187927353e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(6.187927353e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count8 = 0; count8 < generation\_parameter\_3; count8++) {

for (var count7 = 0; count7 < generation\_parameter\_x; count7++) {

for (var count6 = 0; count6 < generation\_parameter\_z; count6++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 145);

if (randomizer <= 10) {

FAD();

} else if (randomizer <= 20) {

iron\_sulphur\_cluster();

} else if (randomizer <= 45) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 10) {

peptidesequence = ' 1 mavlsapglr gfrilglrss vgpavqargv hqsvatdgps stqpalpkar avapkpssrg 61 eyvvaklddl vnwarrsslw pmtfglacca vemmhmaapr ydmdrfgvvf rasprqsdvm 121 ivagtltnkm apalrkvydq mpepryvvsm gscangggyy hysysvvrgc drivpvdiyi 181 pgcpptaeal lygilqlqrk ikrerrlqiw yrr';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 mrclttpmll ralaqaarag ppggrslhss avaatykyvn mqdpemdmks vtdraartll 61 wtelfrglgm tlsylfrepa tinypfekgp lsprfrgeha lrrypsgeer ciacklceai 121 cpaqaitiea epradgsrrt trydidmtkc iycgfcqeac pvdaivegpn fefstethee 181 llynkeklln ngdkweaeia aniqadylyr';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mffsaalrar aagltahwgr hvrnlhktvm qngaggalfv hrdtpennpd tpfdftpeny 61 krieaivkny peghkaaavl pvldlaqrqn gwlpisamnk vaevlqvppm rvyevatfyt 121 mynrkpvgky hiqvctttpc mlrnsdsile aiqkklgikv gettpdklft lieveclgac 181 vnapmvqind nyyedltakd ieeiidelka gkipkpgprs grfscepagg ltslteppkg 241 pgfgvqagl';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 maaaavarlw wrgilgasal trgtgrpsvl llpvrresag adtrptvrpr ndvahkqlsa 61 fgeyvaeilp kyvqqvqvsc fnelevcihp dgvipvltfl rdhtnaqfks lvdltavdvp 121 trqnrfeivy nllslrfnsr irvktytdel tpiesavsvf kaanwyerei wdmfgvffan 181 hpdlrriltd ygfeghpfrk dfplsgyvel ryddevkrvv aepvelaqef rkfdlnspwe 241 afpvyrqppe slkleagdkk pdak';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 maalralcgf rgvaaqvlrp gagvrlpiqp srgvrqwqpd vewaqqfgga vmypsketah 61 wkpppwndvd ppkdtivkni tlnfgpqhpa ahgvlrlvme lsgemvrkcd phigllhrgt 121 eklieyktyl qalpyfdrld yvsmmcneqa yslaveklln irpppraqwi rvlfgeitrl 181 lnhimavtth aldlgamtpf fwlfeerekm fefyervsga rmhaayirpg gvhqdlplgl 241 mddiyqfskn fslrldelee lltnnriwrn rtidigvvta eealnygfsg vmlrgsgiqw 301 dlrktqpydv ydqvefdvpv gsrgdcydry lcrveemrqs lriiaqclnk mppgeikvdd 361 akvsppkrae mktsmeslih hfklytegyq vppgatytai eapkgefgvy lvsdgssrpy 421 rckikapgfa hlagldkmsk ghmladvvai igtqdivfge vdr';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mlatrrllgw slparvsvrf sgdttapkkt sfgslkdedr iftnlygrhd wrlkgslsrg 61 dwyktkeill kgpdwilgei ktsglrgrgg agfptglkws fmnkpsdgrp kylvvnadeg 121 epgtckdrei lrhdphklle gclvggramg araayiyirg efyneasnlq vaireayeag 181 ligknacgsg ydfdvfvvrg agayicgeet aliesiegkq gkprlkppfp advgvfgcpt 241 tvanvetvav spticrrggt wfagfgrern sgtklfnisg hvnhpctvee emsvplkeli 301 ekhaggvtgg wdnllavipg gsstplipks vcetvlmdfd alvqaqtglg taavivmdrs 361 tdivkaiarl iefykhescg qctpcregvd wmnkvmarfv rgdarpaeid slweiskqie 421 ghticalgdg aawpvqglir hfrpeleerm qrfaqqhqar qaas';

protein();

} else if (randomizer <= 70) {

peptidesequence = ' 1 mlripvrkal vglskspkgc vrttataasn lievfvdgqs vmvepgttvl qacekvgmqi 61 prfcyherls vagncrmclv eiekapkvva acampvmkgw niltnseksk karegvmefl 121 lanhpldcpi cdqggecdlq dqsmmfgndr srflegkrav edknigplvk timtrciqct 181 rcirfaseia gvddlgttgr gndmqvgtyi ekmfmselsg niidicpvga ltskpyafta 241 rpwetrktes idvmdavgsn ivvstrtgev mrilprmhed ineewisdkt rfaydglkrq 301 rltepmvrne kglltytswe dalsrvagml qsfqgkdvaa iagglvdaea lvalkdllnr 361 vdsdtlctee vfptagagtd lrsnyllntt iagveeadvv llvgtnprfe aplfnarirk 421 swlhndlkva ligspvdlty tydhlgdspk ilqdiasgsh pfsqvlkeak kpmvvlgssa 481 lqrndgaail aavssiaqki rmtsgvtgdw kvmnilhria sqvaaldlgy kpgveairkn 541 ppkvlfllga dggcitrqdl pkdcfiiyqg hhgdvgapia dvilpgaayt eksatyvnte 601 graqqtkvav tppglaredw kiiralseia gmtlpydtld qvrnrleevs pnlvryddie 661 ganyfqqane lsklvnqqll adplvppqlt ikdfymtdsi srasqtmakc vkavtegaqa 721 veepsic';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mpmanlllli vpiliamafl mlterkilgy mqlrkgpnvv gpygllqpfa damklftkep 61 lkpatstitl yitaptlalt ialllwtplp mpnplvnlnl gllfilatss lavysilwsg 121 wasnsnyali galravaqti syevtlaiil lstllmsgsf nlstlittqe hlwlllpswp 181 lammwfistl aetnrtpfdl aegeselvsg fnieyaagpf alffmaeytn iimmntlttt 241 iflgttydal spelyttyfv tktllltslf lwirtayprf rydqlmhllw knflpltlal 301 lmwyvsmpit issippqt';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 mnplaqpviy stifagtlit alsshwfftw vglemnmlaf ipvltkkmnp rsteaaikyf 61 ltqatasmil lmailfnnml sgqwtmtntt nqysslmimm amamklgmap fhfwvpevtq 121 gtpltsglll ltwqklapis imyqispsln vsllltlsil simagswggl nqtqlrkila 181 yssithmgwm mavlpynpnm tilnltiyii ltttafllln lnsstttlll srtwnkltwl 241 tplipstlls lgglppltgf lpkwaiieef tknnsliipt imatitllnl yfylrliyst 301 sitllpmsnn vkmkwqfeht kptpflptli alttlllpis pfmlmil';

protein();

} else if (randomizer <= 100) {

peptidesequence = ' 1 mnfalilmin tllalllmii tfwlpqlngy mekstpyecg fdpmsparvp fsmkfflvai 61 tfllfdleia lllplpwalq ttnlplmvms sllliiilal slayewlqkg ldwte';

protein();

} else if (randomizer <= 110) {

peptidesequence = ' 1 mlklivptim llpltwlskk hmiwinttth sliisiipll ffnqinnnlf scsptfssdp 61 lttpllmltt wllpltimas qrhlssepls rkklylsmli slqislimtf tatelimfyi 121 ffettliptl aiitrwgnqp erlnagtyfl fytlvgslpl lialiythnt lgslnilllt 181 ltaqelsnsw annlmwlayt mafmvkmply glhlwlpkah veapiagsmv laavllklgg 241 ygmmrltlil npltkhmayp flvlslwgmi mtssiclrqt dlksliayss ishmalvvta 301 iliqtpwsft gavilmiahg ltssllfcla nsnyerthsr imilsqglqt llplmafwwl 361 laslanlalp ptinllgels vlvttfswsn itllltglnm lvtalyslym ftttqwgslt 421 hhinnmkpsf trentlmfmh lspilllsln pdiitgfss';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mpliymniml aftisllgml vyrshlmssl lclegmmlsl fimatlmtln thsllanivp 61 iamlvfaace aavglallvs isntygldyv hnlnllqc';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 mtmhttmttl tltslippil ttlvnpnkkn syphyvksiv astfiislfp ttmfmcldqe 61 viisnwhwat tqttqlslsf kldyfsmmfi pvalfvtwsi mefslwymns dpninqffky 121 lliflitmli lvtannlfql figwegvgim sflliswwya radantaaiq ailynrigdi 181 gfilalawfi lhsnswdpqq mallnanpsl tpllglllaa agksaqlglh pwlpsamegp 241 tpvsallhss tmvvagifll irfhplaens pliqtltlcl gaittlfaav caltqndikk 301 ivafstssql glmmvtigin qphlaflhic thaffkamlf mcsgsiihnl nneqdirkmg 361 gllktmplts tsltigslal agmpfltgfy skdhiietan msytnawals itliatslts 421 aystrmillt ltgqprfptl tninennptl lnpikrlaag slfagflitn nispaspfqt 481 tiplylklta lavtflgllt aldlnyltnk lkmksplctf yfsnmlgfyp sithrtipyl 541 glltsqnlpl llldltwlek llpktisqhq istsiitstq kgmiklyfls fffpliltll 601 lit';

protein();

} else if (randomizer <= 140) {

peptidesequence = ' 1 mmyalfllsv glvmgfvgfs skpspiyggl vlivsgvvgc viilnfgggy mglmvfliyl 61 ggmmvvfgyt tamaieeype awgsgvevlv svlvglamev glvlwvkeyd gvvvvvnfns 121 vgswmiyege gsgliredpi gagalydygr wlvvvtgwtl fvgvyiviei argn';

protein();

} else {

peptidesequence = '1 maaamtfcrl lnrcgeaars lplgarcfgv rvsptgekvt htgqvyddkd yrrirfvgrq 61 kevnenfaid liaeqpvsev etrviacdgg ggalghpkvy inldketktg tcgycglqfr 121 qhhh';

protein();

}

} else if (randomizer <= 50) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 10) {

peptidesequence = ' 1 melvqvlkrg lqqitghggl rgylrvffrt ndakvgtlvg edkygnkyye dnkqffgrhr 61 wvvyttemng kntfwdvdgs mvppewhrwl hsmtddpptt kpltarkfiw tnhkfnvtgt 121 peqyvpystt rkkiqewipp stpyk';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 maavsmsvvl rqtlwrrrav avaalsvsrv ptrslrtstw rlaqdqtqdt qlitvdekld 61 ittltgvpee hiktrkvrif vparnnmqsg vnntkkwkme fdtrerwenp lmgwastadp 121 lsnmvltfst kedavsfaek ngwsydieer kvpkpksksy ganfswnkrt rvstk';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 maaaaqsrvv rvlsmsrsai taiatsvchg ppcrqlhhal mphgkggrss vsgivatvfg 61 atgflgryvv nhlgrmgsqv iipyrcdkyd imhlrpmgdl gqllflewda rdkdsirrvv 121 qhsnvvinli grdwetknfd fedvfvkipq aiaqlskeag vekfihvshl naniksssry 181 lrnkavgekv vrdafpeaii vkpsdifgre drflnsfasm hrfgpiplgs lgwktvkqpv 241 yvvdvskgiv navkdpdang ksfafvgpsr yllfhlvkyi favahrlflp fplplfayrw 301 varvfeispf epwitrdkve rmhitdmklp hlpgledlgi qatplelkai evlrrhrtyr 361 wlsaeiedvk paktvni';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 masrvlsayv srlpaafapl prvrmlavar plstalcsag tqtrlgtlqp alvlaqvpgr 61 vtqlcrqysd mppltlegiq drvlyvlkly dkidpeklsv nshfmkdlgl dsldqveiim 121 amedefgfei pdidaeklmc pqeivdyiad kkdvye';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 maaaaasrgv gaklglreir ihlcqrspgs qgvrdfiekr yvelkkanpd lpilirecsd 61 vqpklwarya fgqetnvpln nfsadqvtra lenvlsgka';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mwfeilpgls vmgvcllipg latayihrft nggkekrvah fgyhwslmer drrisgvdry 61 yvskglenid';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mahehghehg hhkmelpdyr qwkiegtple tiqkklaakg lrdpwgrnea wrymggfaks 61 vsfsdvffkg fkwgfaafvv avgaeyyles lnkdkkhh';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 magvlkkttg lvglavcntp herlrilytk ildvleeipk naayrkyteq itneklamvk 61 aepdvkkled qlqggqleev ilqaehelnl arkmrewklw eplveeppad qwkwpi';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 magsgvrqat stastfvkpi fsrdmneakr rvrelyrawy revpntvhqf qlditvkmgr 61 dkvremfmkn ahvtdprvvd llvikgkiel eetikvwkqr thvmrffhet eaprpkdfls 121 kfyvghdp';

protein();

} else if (randomizer <= 100) {

peptidesequence = ' 1 mapkvfrqyw dipdgtdchr kaysttsias vagltaaayr vtlnppgtfl egvakvgqyt 61 ftaaavgavf glttcisahv rekpddplny flggcagglt lgarthnygi gaaacvyfgi 121 aaslvkmgrl egwevfakpk v';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 maaglfglsa rrllaaaatr glpaarvrwe ssfsrtvvap savagkrppe pttpwqedpe 61 pedenlyekn pdshgydkdp vldvwnmrlv fffgvsiilv lgstfvaylp dyrctgcpra 121 wdgmkewsrr eaerlvkyre anglpimesn cfdpskiqlp ede';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mpfldiqkrf glnidrwlti qsgeqpykma grchafekew iecahgigyt raekeckiey 61 ddfvecllrq ktmrragtir kqrdklikeg kytppphhig kgeprp';

protein();

} else if (randomizer <= 130) {

peptidesequence = '1 msfpkykpss lrtlpetldp aeynispetr raqaerlair aqlkreyllq yndpnrrgli 61 enpallrway artinvypnf rptpknslmg alcgfgplif iyyiikterd rkekliqegk 121 ldrtfhlsy';

protein();

} else if (randomizer <= 140) {

peptidesequence = ' 1 maaskvkqdm pppggygpid ykrnlprrgl sgysmlaigi gtliyghwsi mkwnrerrrl 61 qiedfearia llpllqaetd rrtlqmlren leeeaiimkd vpdwkvgesv fhttrwvppl 121 igelyglrtt eealhashgf mwyt';

protein();

} else {

peptidesequence = ' 1 mgahlvrryl gdasvepdpl qmptfppdyg fperkeremv atqqemmdaq lrlqlrdyca 61 hhlirllkck rdsfpnflac kqerhdwdyc ehrdyvmrmk efererrllq rkkrrekkaa 121 elakgqgpge vdpkval';

protein();

}

} else if (randomizer <= 75) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 10) {

peptidesequence = '1 mpgivelptl eelkvdevki ssavlkaaah hygaqcdkpn kefmlcrwee kdprrcleeg 61 klvnkcaldf frqikrhcae pfteywtcid ytgqqlfrhc rkqqakfdec vldklgwvrp 121 dlgelskvtk vktdrplpen pyhsrprpdp speiegdlqp athgsrfyfw tk';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 maflasgpyl thqqkvlrly kralrhlesw cvqrdkyryf aclmrarfee hknekdmaka 61 tqllkeaeee fwyrqhpqpy ifpdspggts yerydcykvp ewclddwhps ekamypdyfa 121 kreqwkklrr eswerevkql qeetppggpl tealpparke gdlpplwwyi vtrprerpm';

protein();

} else if (randomizer <= 30) {

peptidesequence = '1 mpdswdkdvy pepprrtpvq pnpivymmka fdlivdrpvt lvrefierqh aknryyyyhr 61 qyrrvpdite ckeedimcmy eaemqwkrdy kvdqeiinim qdrlkacqqr egqnyqqnci 121 keveqftqva kayqdryqdl gayssarkcl akqrqrmlqe rkaakeaaaa ts';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mavaragvlg vqwlqrasrn vmplgartas hmtkdmfpgp yprtpeeraa aakkynmrve 61 dyepypddgm gygdypklpd rsqherdpwy swdqpglrln wgepmhwhld mynrnrvdts 121 ptpvswhvmc mqlfgflafm ifmcwvgdvy pvyqpvgpkq ypynnlyler ggdpskeper 181 vvhyei';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 miarrnpepl rflpdearsl pppkltdprl lyigflgycs glidnlirrr piataglhrq 61 llyitafffa gyylvkredy lyavrdremf gymklhpedf peedkktyge ifekfhpir';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 msaltrlasf arvggrlfrs gcartagdgg vrhagggvhi epryrqfpql trsqvfqsef 61 fsglmwfwil wrfwhdseev lghfpypdps qwtdeelgip pdded';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 masatrliqr lrnwasghdl qgklqlryqe iskrtqpppk lpvgpshkls nnyyctrdgr 61 resvppsiim ssqkalvsgk paessavaat ekkavtpapp ikrwelssdq pyl';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 maarvgaflk nawdkepvlv vsfvvgglav ilpplspyfk ysvminkatp ynypvpvrdd 61 gnmpdvpshp qdpqgpslew lkkl';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mlrqiigqak khpsliplfv figtgatgat lyllrlalfn pdvcwdrnnp epwnklgpnd 61 qykfysvnvd ysklkkerpd f';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 maamsllrrv svtavaalsg rplgtrlgfg gfltrgfpka aapvrhsgdh gkrlfvirps 61 rfydrrflkl lrfyialtgi pvaifitlvn vfigqaelae ipegyvpehw eyykhpisrw 121 iarnfydspe kiyertmavl qieaekaelr vkelevrklm hvrgdgpwyy yetidkelid 181 hspkatpdn';

protein();

} else if (randomizer <= 110) {

peptidesequence = ' 1 mvnllqivrd hwvhvlvpmg fvigcyldrk sderltafrn ksmlfkrelq pseevtwk';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mapsallrpl srllaparlp sgpsvrskfy vreppnakpd wlkvgftlgt tvflwiylik 61 qhnedileyk rrngle';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 malrllklaa tsasarvvaa gaqrvrgihs svqcklrygm whfllgdkas krltersrvi 61 tvdgnictgk gklakeiaek lgfkhfpeag ihypdsttgd gkplatdyng ncslekfydd 121 prsndgnsyr lqswlyssrl lqysdalehl lttgqgvvle rsifsdfvfl eamynqgfir 181 kqcvdhynev ksvticdylp phlviyidvp vpevqrriqk kgdphemkit saylqdiena 241 ykktflpems ekcevlqysa reaqdskkvv edieylkfdk gpwlkqdnrt lyhlrllvqd 301 kfevlnytsi piflpevtig ahqtdrvlhq frelpgrkys pgyntevgdk wiwlk';

protein();

} else if (randomizer <= 140) {

peptidesequence = ' 1 magaslgarf yrqikrhpgi ipmigliclg mgsaalyllr lalrspdvcw drknnpepwn 61 rlspndqykf lavstdykkl kkdrpdf';

protein();

} else {

peptidesequence = '1 maapcllrqg ragalktmlq eaqvfrglas tvslsaesgk sekgqpqnsk kqsppknvve 61 pkergkllat qtaaelsknl sspssyppav nkgrkvasps psgsvlftde gvpkflsrkt 121 lvefpqkvls pfrkqgsdse arqvgrkvts psssssssss dsesddeadv sevtprvvsk 181 grgglrkpea shsfenrapr vtvsakektl lqkphvditd pekphqpkkk gspakpsegr 241 enarpkttmp rsqvdeeflk qslkekqlqk tfrlneidke sqkpfevkgp lpvhtksgls 301 appkgspapa vlaeearaeg qlqasppgaa eghlekpvpe pqrkaapplp rketsgtqgi 361 eghlkggqai vedqippsnl etvpvennhg fhektaalkl eaegeameda aapgddrggt 421 qepapvpaep fdnttyknlq hhdystytfl dlnlelskfr mpqpssgres prh';

protein();

}

} else {

randomizer = math\_random\_int(1, 170);

if (randomizer <= 10) {

peptidesequence = '1 mtgytpdekl rlqqlrelrr rwlkdqelsp repvlppqkm gpmekfwnkf lenkspwrkm 61 vhgvykksif vfthvlvpvw iihyymkyhv sekpygivek ksrifpgdti letgevippm 121 kefpdqhh';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 malvhkllrg tyflrkfskp tsalypflgi rfaeyssslq kpvaspgkas sqrktegdlq 61 gdhqkevald itsseekpdv sfdkairdea iyhfrllkde ivdhwrgpeg hplhevlleq 121 akvvwqfrgk edldkwtvts dktiggrsev flkmgknnqs allygtlsse apqdgestrs 181 gycamisrip rgaferkmsy dwsqfntlyl rvrgdgrpwm vnikedtdff qrtnqmysyf 241 mftrggpywq evkipfskff fsnrgrirdv qhelpldkis sigftladkv dgpffleidf 301 igvftdpaht eefayenspe lnprlfk';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mgwsqdlfra lwrslsrevk ehvgtdqfgn kyyyipqykn wrgqtirekr iveaankkev 61 dyeagdipte weawirrtrk tpptmeeilk nekhreeiki ksqdfyekek llsketseel 121 lpppvqtqik ghasapyfgk eepsvapsst gktfqpgswm prdgkshnq';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 matalalrsl yrarpslrcp pvelpwaprr ghrlspadde lyqrtrisll qreaaqamyi 61 dsynsrgfmi ngnrvlgpca llphsvvqwn vgshqdited sfslfwllep rieivvvgtg 121 drterlqsqv lqamrqrgia vevqdtpnac atfnflcheg rvtgaalipp pggtsltslg 181 qaaq';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mgalvirgir nfnlenraer eiskmkpsva prhpstnsll reqislypev kgeiarkdek 61 llsflkdvyv dskdpvsslq vkaaetcqep kefrlpkdhh fdminiksip kgkisiveal 121 tllnnhklfp etwtaekimq eyqleqkdvn sllkyfvtfe veifppedkk airsk';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 msgcglflrt taaaracrgl vvstanrrll rtsppvrafa kelflgkikk kevfpfpevs 61 qdelneinqf lgpvekffte evdsrkidqe gkipdetlek lkslglfglq vpeeygglgf 121 sntmysrlge iismdgsitv tlaahqaigl kgiilagtee qkakylpkla sgehiaafcl 181 tepasgsdaa sirsratlse dkkhyilngs kvwitnggla niftvfakte vvdsdgsvkd 241 kitafiverd fggvtngkpe dklgirgsnt cevhfentki pvenilgevg dgfkvamnil 301 nsgrfsmgsv vagllkrlie mtaeyactrk qfnkrlsefg liqekfalma qkayvmesmt 361 yltagmldqp gfpdcsieaa mvkvfsseaa wqcvsealqi lgglgytrdy pyerilrdtr 421 illifegtne ilrmyialtg lqhagriltt rihelkqakv stvmdtvgrr lrdslgrtvd 481 lgltgnhgvv hpsladsank feentycfgr tvetlllrfg ktimeeqlvl krvanilinl 541 ygmtavlsra srsiriglrn hdhevllant fcveaylqnl fslsqldkya penldeqikk 601 vsqqilekra yicahpldrt c';

protein();

} else if (randomizer <= 70) {

peptidesequence = ' 1 mswvqatlla rglcrawggt cgaaltgtsi sqvprrlprg lhcsaaahss eqslvpsppe 61 prqrptkalv pfedlfgqap ggerdkasfl qtvqkfaehs vrkrghidfi ylalrkmrey 121 gverdlavyn qllnifpkev frprniiqri fvhyprqqec giavleqmen hgvmpnkete 181 flliqifgrk sypmlklvrl klwfprfmnv npfpvprdlp qdpvelamfg lrhmepdlsa 241 rvtiyqvplp kdstgaadpp qphivgiqsp dqqaalarhn parpvfvegp fslwlrnkcv 301 yyhilradll ppeereveet peewnlyypm qldleyvrsg wdnyefdine veegpvfamc 361 magahdqatm akwiqglqet nptlaqipvv frlagstrel qtssagleep plpedhqeed 421 dnlqrqqqgq s';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mirrvlphgm grglltrrpg trrggfsldw dgkvseikkk iksilpgrsc dllqdtshlp 61 pehsdvvivg ggvlglsvay wlkklesrrg airvlvverd htysqastgl svggicqqfs 121 lpeniqlslf sasflrnine ylavvdappl dlrfnpsgyl llasekdaaa mesnvkvqrq 181 egakvslmsp dqlrnkfpwi ntegvalasy gmedegwfdp wcllqglrrk vqslgvlfcq 241 gevtrfvsss qrmlttddka vvlkrihevh vkmdrsleyq pvecaivina agawsaqiaa 301 lagvgegppg tlqgtklpve prkryvyvwh cpqgpgletp lvadtsgayf rreglgsnyl 361 ggrspteqee pdpanlevdh dffqdkvwph lalrvpafet lkvqsawagy ydyntfdqng 421 vvgphplvvn myfatgfsgh glqqapgigr avaemvlkgr fqtidlspfl ftrfylgeki 481 qennii';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mvvfgyeagt kprdsgvvpv gteeapkvfk maasmhgqps psledaklrr pmvieiiekn 61 fdylrkemtq niyqmatfgt tagfsgifsn flfrrcfkvk hdalktyasl atlpflstvv 121 tdklfvidal ysdniskenc vfrssligiv cgvfypssla ftkngrlatk yhtvplppkg 181 rvlihwmtlc qtqmklmaip lvfqimfgil nglyhyavfe etlektihee';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mevpppaprs flcralclfp rvfaaeavta dsevleerqk rlpyvpepyy pesgwdrlre 61 lfgkdeqqri skdlanickt aatagiigwv yggipafiha kqqyieqsqa eiyhnrfdav 121 qsahraatrg firygwrwgw rtavfvtifn tvntslnvyr nkdalshfvi agavtgslfr 181 invglrglva ggiigallgt pvggllmafq kysgetvqer kqkdrkalhe lkleewkgrl 241 qvtehlpeki esslqedepe ndakkieall nlprnpsvid kqdkd';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 mwdlvledrr mnslrlvapm wngrirgihr lgaavapegn qkkkrtilqf ltnyfydvea 61 lrdyllqrem ykvheknrsy twlekqhgpy gagaffilkq ggavkfrdke wirpdkyghf 121 sqefwnfcev pveavdagdc dinyegldnl lrlkelqsls lqrcchvddw clsrlyplad 181 slqelslagc priserglac lhhlqnlrrl disdlpavsn pgltqilvee mlpncevvgv 241 dwaeglksgp eeqprdtasp vpa';

protein();

} else if (randomizer <= 120) {

peptidesequence = '1 matalseeel dnedyyslln vrreasseel kaayrrlcml yhpdkhrdpe lksqaerlfn 61 lvhqayevls dpqtraiydi ygkrglemeg wevverrrtp aeireeferl qrereerrlq 121 qrtnpkgtis vgvdatdlfd rydeeyedvs gssfpqiein kmhisqsiea pltatdtail 181 sgslstqngn gggsinfalr rvtsakgwge lefgagdlqg plfglklfrn ltprcfvttn 241 calqfssrgi rpglttvlar nldkntvgyl qwrwgiqsam ntsivrdtkt shftvalqlg 301 iphsfalisy qhkfqdddqt rvkgslkagf fgtvveygae rkisrhsvlg aavsvgvpqg 361 vslkvklnra sqtyffpihl tdqllpsamf yatvgplvvy famhrliikp ylraqkekel 421 ekqresaatd vlqkkqeaes avrlmqesvr riieaeesrm gliivnawyg kfvndksrks 481 ekvkvidvtv plqclvkdsk lilteaskag lpgfydpcvg eeknlkvlyq frgvlhqvmv 541 ldsealripk qshridtdg';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 menhksnnke nitivdisrk inqlpeaern llengsvyvg lnaalcglia nslfrrilnv 61 tkariaaglp magipflttd ltyrcfvsfp lntgdldcet ctitrsgltg lvigglypvf 121 laipvnggla aryqsallph kgnilsywir tskpvfrkml fpillqtmfs aylgseqykl 181 likalqlsep gkeih';

protein();

} else if (randomizer <= 140) {

peptidesequence = '1 misasraaaa rlvgaaasrg ptaarhqdsw nglsheafrl vsrrdyasea ikgavvgidl 61 gttnscvavm egkqakvlen aegarttpsv vaftadgerl vgmpakrqav tnpnntfyat 121 krligrrydd pevqkdiknv pfkivrasng dawveahgkl yspsqigafv lmkmketaen 181 ylghtaknav itvpayfnds qrqatkdagq isglnvlrvi neptaaalay gldksedkvi 241 avydlgggtf disileiqkg vfevkstngd tflggedfdq allrhivkef kretgvdltk 301 dnmalqrvre aaekakcels ssvqtdinlp yltmdssgpk hlnmkltraq fegivtdlir 361 rtiapcqkam qdaevsksdi gevilvggmt rmpkvqqtvq dlfgrapska vnpdeavaig 421 aaiqggvlag dvtdvllldv tplslgietl ggvftklinr nttiptkksq vfstaadgqt 481 qveikvcqge remagdnkll gqftligipp aprgvpqiev tfdidangiv hvsakdkgtg 541 reqqiviqss gglskddien mvknaekyae edrrkkerve avnmaegiih dtetkmeefk 601 dqlpadecnk lkeeiskmre llarkdsetg enirqaassl qqaslklfem aykkmasere 661 gsgssgtgeq kedqkeekq';

protein();

} else if (randomizer <= 150) {

peptidesequence = ' 1 mlaarlvclr tlpsrvfhpa ftkaspvvkn sitknqwllt psreyatktr igirrgrtgq 61 elkeaaleps mekifkidqm grwfvaggaa vglgalcyyg lglsneigai ekaviwpqyv 121 kdrihstymy lagsigltal saiaisrtpv lmnfmmrgsw vtigvtfaam vgagmlvrsi 181 pydqspgpkh lawllhsgvm gavvapltil ggplliraaw ytagivggls tvamcapsek 241 flnmgaplgv glglvfvssl gsmflppttv agatlysvam ygglvlfsmf llydtqkvik 301 raevspmygv qkydpinsml siymdtlnif mrvatmlatg gnrkk';

protein();

} else if (randomizer <= 160) {

peptidesequence = ' 1 mlrpaglwrl crrpwaarvp aenlgrrevt sgvsprgsts prtlnifdrd lkrkqknwaa 61 rqpeptkfdy lkeevgsria drvydiprnf plaldlgcgr gyiaqylnke tigkffqadi 121 aenalknsse teiptvsvla deeflpfken tfdlvvssls lhwvndlpra leqihyilkp 181 dgvfigamfg gdtlyelrcs lqlaetereg gfsphispft avndlghllg ragfntltvd 241 tdeiqvnypg mfelmedlqg mgesncawnr kallhrdtml aaaavyremy rnedgsvpat 301 yqiyymigwk yhesqarpae rgsatvsfge lgkinnlmpp gkksq';

protein();

} else {

peptidesequence = ' 1 msvllrsglg plcavaraai pfiwrgkyfs sgnepaenpv tpmlrhlmyk ikstgpitva 61 eymkevltnp akgyyvyrdm lgekgdfits peisqifgel lgiwfisewm atgkstafql 121 velgpgrgtl vgdilrvftq lgsvlkncdi svhlvevsqk lseiqaltlt kekvplerna 181 gspvymkgvt ksgipiswyr dlhdvpkgys fylaheffdv lpvhkfqktp qgwrevfvdi 241 dpqvsdklrf vlapsatpae afiqhdetrd hvevcpdagv iieelsqria ltggaalvad 301 yghdgtktdt frgfcdhklh dvliapgtad ltadvdfsyl rrmaqgkvas lgpikqhtfl 361 knmgidvrlk vlldksneps vrqqllqgyd mlmnpkkmge rfnffallph qrlqggryqr 421 narqskpfas vvagfselaw q';

protein();

}

}

} else if (randomizer <= 70) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 10) {

peptidesequence = '1 msgvrglsrl lsarrlalak awptvlqtgt rgfhftvdgn krasakvsds isaqypvvdh 61 efdavvvgag gaglraafgl seagfntacv tklfptrsht vaaqgginaa lgnmeednwr 121 whfydtvkgs dwlgdqdaih ymteqapaav velenygmpf srtedgkiyq rafggqslkf 181 gkggqahrcc cvadrtghsl lhtlygrslr ydtsyfveyf aldllmenge crgvialcie 241 dgsihrirak ntvvatggyg rtyfsctsah tstgdgtami traglpcqdl efvqfhptgi 301 ygagcliteg crgeggilin sqgerfmery apvakdlasr dvvsrsmtle iregrgcgpe 361 kdhvylqlhh lppeqlatrl pgisetamif agvdvtkepi pvlptvhynm ggiptnykgq 421 vlrhvngqdq ivpglyacge aacasvhgan rlganslldl vvfgracals ieescrpgdk 481 vppikpnage esvmnldklr fadgsirtse lrlsmqksmq nhaavfrvgs vlqegcgkis 541 klygdlkhlk tfdrgmvwnt dlvetlelqn lmlcalqtiy gaearkesrg aharedykvr 601 ideydyskpi qgqqkkpfee hwrkhtlsyv dvgtgkvtle yrpvidktln eadcatvppa 661 irsy';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mpgrhvsrvr alykrvlqlh rvlppdlksl gdqyvkdefr rhktvgsdea qrflqewevy 61 atallqqane nrqnstgkac fgtflpeekl ndfrdeqigq lqelmqeatk pnrqfsises 121 mkpkf';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 maavvalslr rrlpattlgg aclqasrgaq taaataprik kfaiyrwdpd kagdkphmqt 61 yevdlnkcgp mvldalikik nevdstltfr rscregicgs camninggnt lactrridtn 121 lnkvskiypl phmyvikdlv pdlsnfyaqy ksiepylkkk desqegkqqy lqsieerekl 181 dglyecilca ccstscpsyw wngdkylgpa vlmqayrwmi dsrddfteer laklqdpfsl 241 yrchtimnct rtcpkglnpg kaiaeikkmm atykekkasv';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mtpsrlpwll swvsatawra arspllchsl rktsssqggk selvkqslkk pklpegrfda 61 pedshlekep lekfpddvnp vtkekggprg peptrygdwe rkgrcidf';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 maalllrhvg rhclrahfsp qlcirnavpl gttakeemer fwnknigsnr plsphitiys 61 wslpmamsic hrgtgialsa gvslfgmsal llpgnfesyl elvkslclgp alihtakfal 121 vfplmyhtwn girhlmwdlg kglkipqlyq sgvvvlvltv lssmglaam';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 maalllrhvg rhclrahfsp qlcirnavpl gttakeemer fwnknigsnr plsphitiys 61 wslpmamsic hrgtgialsa gvslfgmsal llpgnfesyl elvkslclgp alihtakfal 121 vfplmyhtwn girhlmwdlg kglkipqlyq sgvvvlvltv lssmglaam';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mavlwrlsav cgalggrall lrtpvvrpah isaflqdrpi pewcgvqhih lspshhsgsk 61 aaslhwtser vvsvlllgll paaylnpcsa mdyslaaalt lhghwglgqv vtdyvhgdal 121 qkaakaglla lsaltfaglc yfnyhdvgic kavamlwkl';

protein();

} else if (randomizer <= 80) {

peptidesequence = '1 mavlwrlsav cgalggrall lrtpvvrpah isaflqdrpi pewcgvqhih lspshhsgsk 61 aaslhwtser vvsvlllgll paaylnpcsa mdyslaaalt lhghwglgqv vtdyvhgdal 121 qkaakaglla lsaltfaglc yfnyhdvgic kavamlwkl';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 mavstvfsts slmlalsrhs llspllsvts frrfyrgdsp tdsqkdmiei plppwqertd 61 esietkrarl lyesrkrgml encillslfa kehlqhmtek qlnlydrlin epsndwdiyy 121 wateakpape ifenevmall rdfaknknke qrlrapdley lfekpr';

protein();

} else {

peptidesequence = ' 1 msrhsrlqrq vlslyrdllr agrgkpgaea rvraefrqha glprsdvlri eylyrrgrrq 61 lqllrsghat amgafvrpra ptgepggvgc qpddgdsprn phdstgapet rpdgr';

protein();

}

} else if (randomizer <= 95) {

randomizer = math\_random\_int(1, 150);

if (randomizer <= 10) {

peptidesequence = ' 1 mtpmrktnpl mklinhsfid lptpsnisaw wnfgsllgac lilqittglf lamhyspdas 61 tafssiahit rdvnygwiir ylhangasmf ficlflhigr glyygsflys etwnigiill 121 latmatafmg yvlpwgqmsf wgatvitnll saipyigtdl vqwiwggysv dsptltrfft 181 fhfilpfiia alatlhllfl hetgsnnplg itshsdkitf hpyytikdal glllfllslm 241 tltlfspdll gdpdnytlan plntpphikp ewyflfayti lrsvpnklgg vlalllsili 301 lamipilhms kqqsmmfrpl sqslywllaa dlliltwigg qpvsypftii gqvasvlyft 361 tililmptis lienkmlkwa';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 maaaaaslrg vvlgprgagl pgarargllc sarpgqlplr tpqavalssk sglsrgrkvm 61 lsalgmlaag gaglamalhs avsasdlelh ppsypwshrg llssldhtsi rrgfqvykqv 121 caschsmdfv ayrhlvgvcy tedeakelaa evevqdgpne dgemfmrpgk lfdyfpkpyp 181 nseaaraann galppdlsyi vrarhggedy vfslltgyce pptgvslreg lyfnpyfpgq 241 aiamappiyt dvlefddgtp atmsqiakdv ctflrwasep ehdhrkrmgl kmlmmmallv 301 plvytikrhk wsvlksrkla yrppk';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mlsvasrsgp fapvlsatsr gvagalrplv qatvpatpeq pvldlkrpfl sreslsgqav 61 rrplvasvgl nvpasvcysh tdikvpdfse yrrlevldst kssressear kgfsylvtgv 121 ttvgvayaak navtqfvssm sasadvlala kieiklsdip egknmafkwr gkplfvrhrt 181 qkeieqeaav elsqlrdpqh dldrvkkpew viligvcthl gcvpianagd fggyycpchg 241 shydasgrir lgpaplnlev ptyeftsddm vivg';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 maasvvcraa tagaqvllra rrspallrtp alrstatfaq alqfvpetqv slldnglrva 61 seqssqptct vgvwidvgsr feteknngag yflehlafkg tknrpgsale kevesmgahl 121 naystrehta yyikalskdl pkavellgdi vqncsledsq iekerdvilr emqendasmr 181 dvvfnylhat afqgtplaqa vegpsenvrk lsradlteyl sthykaprmv laaaggvehq 241 qlldlaqkhl ggipwtyaed avptltpcrf tgseirhrdd alpfahvaia vegpgwaspd 301 nvalqvanai ighydctygg gvhlssplas gavanklcqs fqtfsicyae tgllgahfvc 361 drmkiddmmf vlqgqwmrlc tsatesevar gknilrnalv shldgttpvc edigrsllty 421 grriplaewe sriaevdasv vreicskyiy dqcpavagyg pieqlpdynr irsgmfwlrf';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mklltragsf srfyslkvap kvkataapag appqpqdlef tklpnglvia slenyspvsr 61 iglfikagsr yedfsnlgtt hllrltsslt tkgassfkit rgieavggkl svtatrenma 121 ytveclrgdv dilmefllnv ttapefrrwe vadlqpqlki dkavafqnpq thvienlhaa 181 ayrnalanpl ycpdyrigkv tseelhyfvq nhftsarmal iglgvshpvl kqvaeqflnm 241 rgglglsgak anyrggeire qngdslvhaa fvaesavags aeanafsvlq hvlgagphvk 301 rgsnttshlh qavakatqqp fdvsafnasy sdsglfgiyt isqataagdv ikaaynqvkt 361 iaqgnlsntd vqaaknklka gylmsvesse cfleevgsqa lvagsympps tvlqqidsva 421 nadiinaakk fvsgqksmaa sgnlghtpfv del';

protein();

} else if (randomizer <= 60) {

peptidesequence = '1 mgledeqkml tesgdpeeee eeeeelvdpl ttvreqceql ekcvkarerl elcdervssr 61 shteedctee lfdflhardh cvahklfnnl k';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 magkqavsas gkwldgirkw yynaagfnkl glmrddtiye dedvkeairr lpenlyndrm 61 frikraldln lkhqilpkeq wtkyeeenfy lepylkevir erkereewak k';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mgrefgnltr mrhvisysls pfeqrayphv ftkgipnvlr riresffrvv pqfvvfyliy 61 twgteefers krknpaayen dk';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 maaatltskl ysllfrrtst faltiivgvm fferafdqga daiydhineg klwkhikhky 61 enk';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 mvtrflgpry relvknwvpt aytwgavgav glvwatdwrl ildwvpying kfkkdn';

protein();

} else if (randomizer <= 110) {

peptidesequence = ' 1 mfrllswslg rgflraagrr crgcsarllp glaggpgpev qvppsrvaph grgpgllpll 61 aalawfsrpa aaeeeeqqga dgaaaedgad eaeaeiiqll kraklsimkd epeeaelilh 121 dalrlayqtd nkkaitytyd lmanlafirg qlenaeqlfk atmsyllggg mkqednaiie 181 islklasiya aqnrqefava gyefcistle ekierekela edimsveeka nthlllgmcl 241 dacaryllfs kqpsqaqrmy ekalqiseei qgerhpqtiv lmsdlattld aqgrfdeayi 301 ymqrasdlar qinhpelhmv lsnlaavlmh rerytqakei yqealkqakl kkdeisvqhi 361 reelaelskk srpltnsvkl';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mkdepeeael ilhdalrlay qtdnkkaity tydlmanlaf irgqlenaeq lfkatmsyll 61 gggmkqedna iieislklas iyaaqnrqef avagyefcis tleekierek elaedimsve 121 ekanthlllg mcldacaryl lfskqpsqaq rmyekalqis eeiqgerhpq tivlmsdlat 181 tldaqgrfde ayiymqrasd larqinhpel hmvlsnlaav lmhrerytqa keiyqealkq 241 aklkkdeisv qhireelael skksrpltns vkl';

protein();

} else if (randomizer <= 130) {

peptidesequence = '1 mgravkvlql fktlhrtrqq vfkndarale aarikineef knnksetssk kieelmkigs 61 dvelllrtsv iqgihtdhnt lklvprkdll venvpycdap tqkq';

protein();

} else if (randomizer <= 140) {

peptidesequence = '1 maqiitrmcc yaigvnsgle hnvgsshfcl dsawtdekqv eksevdfsks hslvrrfedl 61 kpklsvcktg sqvfrsenwk vwaessrgdh ddcldlcsvl cwgellqtip eippkrgelk 121 tellglkerk hkpqvsqqee lk';

protein();

} else {

peptidesequence = ' 1 mpagvpmsty lkmfaaslla mcagaevvhr yyrpdltipe ippkrgelkt ellglkerkh 61 kpqvsqqeel k';

protein();

}

} else if (randomizer <= 120) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 50) {

randomizer = math\_random\_int(1, 200);

if (randomizer <= 10) {

peptidesequence = '1 mfadrwlfst nhkdigtlyl lfgawagvlg talsllirae lgqpgnllgn dhiynvivta 61 hafvmiffmv mpimiggfgn wlvplmigap dmafprmnnm sfwllppsll lllasamvea 121 gagtgwtvyp plagnyshpg asvdltifsl hlagvssilg ainfittiin mkppamtqyq 181 tplfvwsvli tavllllslp vlaagitmll tdrnlnttff dpagggdpil yqhlfwffgh 241 pevyililpg fgmishivty ysgkkepfgy mgmvwammsi gflgfivwah hmftvgmdvd 301 trayftsatm iiaiptgvkv fswlatlhgs nmkwsaavlw algfiflftv ggltgivlan 361 ssldivlhdt yyvvahfhyv lsmgavfaim ggfihwfplf sgytldqtya kihftimfig 421 vnltffpqhf lglsgmprry sdypdayttw nilssvgsfi sltavmlmif miweafaskr 481 kvlmveepsm nlewlygcpp pyhtfeepvy mks';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 mahaaqvglq datspimeel itfhdhalmi iflicflvly alfltlttkl tntnisdaqe 61 metvwtilpa iilvlialps lrilymtdev ndpsltiksi ghqwywtyey tdygglifns 121 ymlpplflep gdlrlldvdn rvvlpieapi rmmitsqdvl hswavptlgl ktdaipgrln 181 qttftatrpg vyygqcseic ganhsfmpiv leliplkife mgpvftl';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mthqshayhm vkpspwpltg alsallmtsg lamwfhfhsm tllmlglltn tltmyqwwrd 61 vtrestyqgh htppvqkglr ygmilfitse vfffagffwa fyhsslaptp qlgghwpptg 121 itplnplevp llntsvllas gvsitwahhs lmennrnqmi qallitillg lyftllqase 181 yfespftisd giygstffva tgfhglhvii gstflticfi rqlmfhftsk hhfgfeaaaw 241 ywhfvdvvwl flyvsiywwg s';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mlatrvfslv gkraistsvc vrahesvvks edfslpaymd rrdhplpeva hvkhlsasqk 61 alkekekasw sslsmdekve lyrikfkesf aemnrgsnew ktvvggamff igftalvimw 121 qkhyvygplp qsfdkewvak qtkrmldmkv npiqglaskw dyeknewkk';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mlpraawslv lrkggggrrg mhssegttrg ggkmspytnc yaqryypmpe epfctelnae 61 eqalkekekg swtqlthaek valyrlqfne tfaemnrrsn ewktvmgcvf ffigfaalvi 121 wwqrvyvfpp kpitltderk aqqlqrmldm kvnpvqglas rwdyekkqwk k';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mlgaalrrca vaattradpr gllhsartpg pavaiqsvrc yshgsqetde efdarwvtyf 61 nkpdidawel rkgintlvty dmvpepkiid aalracrrln dfastvrile vvkdkagphk 121 eiypyviqel rptlnelgis tpeelgldkv';

protein();

} else if (randomizer <= 70) {

peptidesequence = ' 1 masrllrgag tlaaqalrar gpsgaaamrs masgggvptd eeqatglere imlaakkgld 61 pynvlapkga sgtredpnlv psisnkrivg ciceedntsv vwfwlhkgea qrcprcgahy 121 klvpqqlah';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mavvgvssvs rllgrsrpql grpmssgahg eegsarmwkt ltffvalpgv avsmlnvylk 61 shhgeherpe fiayphlrir tkpfpwgdgn htlfhnphvn plptgyede';

protein();

} else if (randomizer <= 90) {

peptidesequence = ' 1 malplrpltr glasaakggh ggagartwrl ltfvlalpsv alctfnsylh sghrprpefr 61 pyqhlrirtk pypwgdgnht lfhnshvnpl ptgyehp';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 maedmetkik nyktapfdsr fpnqnqtrnc wqnyldfhrc qkamtakggd isvcewyqrv 61 yqslcptswv tdwdeqraeg tfpgki';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 mldveaqepp kgkwstppfd prfpsqnqir ncyqnfldyh rclktrtrrg kstqpceyyf 61 rvyhslcpis wveswneqik ngifagki';

protein();

} else if (randomizer <= 120) {

peptidesequence = '1 mapevlpkpr mrgllarrlr nhmavafvls lgvaalykfr vadqrkkaya dfyrnydvmk 61 dfeemrkagi fqsvk';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 mqalrvsqal irsfsstarn rfqnrvrekq klfqedndip lylkggivdn ilyrvtmtlc 61 lggtvyslys lgwasfprn';

protein();

} else if (randomizer <= 140) {

peptidesequence = ' 1 mlrnllalrq igqrtistas rrhfknkvpe kqklfqedde iplylkggva dallyratmi 61 ltvggtayai yelavasfpk kqe';

protein();

} else if (randomizer <= 150) {

peptidesequence = ' 1 mlwnllalhq igqrtistas hrhfknkvpe kqklfqeddg iplylkggia dallhratmi 61 ltvggtayai yqlavasfpn kgvtsiipai twftfiqlsm dqksdk';

protein();

} else if (randomizer <= 160) {

peptidesequence = ' 1 mfplvksaln rlqvrsiqqt marqshqkrt pdfhdkygna vlasgatfci vtwtyvatqv 61 giewnlspvg rvtpkewrnq';

protein();

} else if (randomizer <= 170) {

peptidesequence = ' 1 mlgqsirrft tsvvrrshye egpgknlpfs venkwsllak mclyfgsafa tpflvvrhql 61 lkt';

protein();

} else if (randomizer <= 180) {

peptidesequence = ' 1 myykfsgftq klagawasea yspqglkpvv steappiifa tptkltsdst vydyagknkv 61 pelqkffqka dgvpvylkrg lpdqmlyrtt maltvggtiy clialymasq pknk';

protein();

} else if (randomizer <= 190) {

peptidesequence = ' 1 msvltplllr gltgsarrlp vprakihslp pegklgimel avgltscfvt fllpagwils 61 hletyrrpe';

protein();

} else {

peptidesequence = '1 mpllrgrcpa rrhyrrlall glqpaprfah sgpprqrpls aaemavglvv ffttfltpaa 61 yvlgnlkqfr rn';

protein();

}

} else {

randomizer = math\_random\_int(1, 200);

if (randomizer <= 10) {

peptidesequence = ' 1 mmwqkyagsr rsmplgaril fhgvfyaggf aivyyliqkf hsralyykla veqlqshpea 61 qealgpplni hylklidren fvdivdaklk ipvsgskseg llyvhssrgg pfqrwhldev 121 flelkdgqqi pvfklsgeng devkke';

protein();

} else if (randomizer <= 20) {

peptidesequence = ' 1 massgagdpl dskrgeapfa qridptrekl tpeqlhsmrq aelaqwqkvl prrrtrnivt 61 glgigalvla iygytfysis qerfldeled eakaararal arasgs';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mstsvpqght wtqrvkkdde eedpldqlis rsgcaashfa vqecmaqhqd wrqcqpqvqa 61 fkdcmseqqa rrqeelqrrq eqagahh';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 mpkyyedkpq ggacaglked lgacllqsdc vvqegksprq clkegycnsl kyaffeckrs 61 vldnrarfrg rkgy';

protein();

} else if (randomizer <= 50) {

peptidesequence = '1 mgpggpllsp srgfllcktg whsnrllgdc gphtpvstal sfiavgmaap smkerqvcwg 61 ardeywkcld enledasqck klrssfessc pqqwikyfdk rrdylkfkek feagqfepse 121 ttaks';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 magmvdfqde eqvksflenm evecnyhcyh ekdpdgcyrl vdylegirkn fdeaakvlkf 61 nceenqhsds cyklgayyvt gkggltqdlk aaarcflmac ekpgkksiaa chnvgllahd 121 gqvnedgqpd lgkardyytr acdggytssc fnlsamflqg apgfpkdmdl ackysmkacd 181 lghiwacana srmyklgdgv dkdeakaevl knraqqlhke qqkgvqpltf g';

protein();

} else if (randomizer <= 70) {

peptidesequence = ' 1 mgglwrpgwr cvpfcgwrwi hpgsptraae rvepflrpew sgtggaergl rwlgtwkrcs 61 lrarhpalqp prrpkssnpf traqeeerrr qnkttltyva avavgmlgas yaavplyrly 121 cqttglggsa vaghasdkie nmvpvkdrii kisfnadvha slqwnfrpqq teiyvvpget 181 alafyraknp tdkpvigist ynivpfeagq yfnkiqcfcf eeqrlnpqee vdmpvffyid 241 pefaedprmi kvdlitlsyt ffeakeghkl pvpgyn';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mptgkqladi gyktfstsmm lltvyggylc svrvyhyfqw rraqrqaaee qktsgim';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mqrllfpplr alkgrqylpl lapraapraq cdcirrplrp gqystiseva lqsgrgtvsl 61 pskaaervvg rwllvcsgtv agavilggvt rltesglsmv dwhlikemkp ptsqeeweae 121 fqryqqfpef kilnhdmtlt efkfiwymey shrmwgrlvg lvyilpaayf wrkgwlsrgm 181 kgrvlalcgl vcfqgllgwy mvksgleeks dshdiprvsq yrlaahlgsa lvlycaslwt 241 slslllpphk lpethqllql rrfahgtagl vfltalsgaf vagldaglvy nsfpkmgesw 301 ipedlftfsp ilrnvfenpt mvqfdhrilg itsvtaitvl yflsrriplp rrtkmaavtl 361 lalaytqvgl gistllmyvp tplaathqsg slalltgalw lmnelrrvpk';

protein();

} else if (randomizer <= 100) {

peptidesequence = ' 1 mfapavmraf rknktlgygv pmlllivggs fglrefsqir ydavkskmdp elekklkenk 61 isleseyeki kdskfddwkn irgprpwedp dllqgrnpes lktktt';

protein();

} else if (randomizer <= 110) {

peptidesequence = '1 mpglvdsnpa ppesqekkpl kpccacpetk kardawfrgr vsvdlsrslv pgndgerger 61 qastthsflq iiiekgeehc ghlieahkec mralgfki';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mlcrlggrwl rplpalqlwa rdlplapvpt sgakrptlpv wavapvsavh angwyealaa 61 sspvrvaeev llgvhaatgl pwwgsillst valrgavtlp laayqhyila kvenlqpeik 121 tiarhlnqev avranqlgws krdarltylk nmrrlisely vrdnchpfka tvlvwiqlpm 181 wifmsfalrn lstgaahsea gfsvqeqlat ggilwfpdlt apdstwilpi svgvinlliv 241 eicalqkigm srfqtyityf vramsvlmip iaatvpssiv lywlcssfvg lsqnlllrsp 301 gfrqlcrips tksdsetpyk difaafntkf isrk';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 mstamnfgtk sfqprppdkg sfpldhlgec ksfkekfmkc lhnnnfenal crkeskeyle 61 crmerklmlq epleklgfgd ltsgkseakk';

protein();

} else if (randomizer <= 140) {

peptidesequence = ' 1 maappepgep eerkslkllg fldventpca rhsilygslg svvagfghfl ftsrirrscd 61 vgvggfilvt lgcwfhcryn yakqriqeri areeikkkil yegthldper khngsssn';

protein();

} else if (randomizer <= 150) {

peptidesequence = ' 1 maavaalqlg lraaglgrap asaawrsvlr vsprpgvawr psrcgssaae asatkaedds 61 flqwvlllip vtafglgtwq vqrrkwklnl iaelesrvla epvplpadpm elknleyrpv 121 kvrgcfdhsk elymmprtmv dpvrearegg lissstqsga yvvtpfhctd lgvtilvnrg 181 fvprkkvnpe trqkgqiege vdligmvrlt etrqpfvpen npernhwhyr dleamaritg 241 aepifidanf qstvpggpig gqtrvtlrne hlqyivtwyg lsaatsylwf kkflrgtpgv';

protein();

} else if (randomizer <= 160) {

peptidesequence = '1 mamlvlvpgr vmrplggqlw rflprglefw gpaegtarvl lrqfcarqae awrasgrpgy 61 clgtrplsta rppppwsqkg pgdstrpskp gpvswkslai tfaiggalla gmkhvkkeka 121 eklekerqrh igkpllggpf sltthtgerk tdkdylgqwl liyfgfthcp dvcpeelekm 181 iqvvdeidsi ttlpdltplf isidperdtk eaianyvkef spklvgltgt reevdqvara 241 yrvyyspgpk dededyivdh tiimyligpd gefldyfgqn krkgeiaasi athmrpyrkk 301 s';

protein();

} else if (randomizer <= 170) {

peptidesequence = ' 1 mllltrspta whrlsqlkpr vlpgtlggqa lhlrswllsr qgpaetggqg qpqgpglrtr 61 llitglfgag lggawlalra ekerlqqqkr tealrqaavg qgdfhlldhr grarckadfr 121 gqwvlmyfgf thcpdicpde leklvqvvrq leaepglppv qpvfitvdpe rddveamary 181 vqdfhprllg ltgstkqvaq ashsyrvyyn agpkdedqdy ivdhsiaiyl lnpdglftdy 241 ygrsrsaeqi sdsvrrhmaa frsvls';

protein();

} else if (randomizer <= 180) {

peptidesequence = ' 1 maallrsarw llragaaprl plslrllpgg pgrlhaasyl paaragpvag gllsparlya 61 iaakekdiqe estfssrkis nqfdwalmrl dlsvrrtgri pkkllqkvfn dtcrsgglgg 121 shallllrsc gsllpelkle ertefahriw dtlqklgavy dvshynallk vylqneykfs 181 ptdflakmee aniqpnrvty qrliasycnv gdiegaskil gfmktkdlpv teavfsalvt 241 gharagdmen aeniltvmrd agiepgpdty lallnayaek gdidhvkqtl ekvekselhl 301 mdrdllqiif sfskagypqy vseilekvtc erryipdamn lilllvtekl edvalqilla 361 cpvskedgps vfgsfflqhc vtmntpvekl tdyckklkev qmhsfplqft lhcallankt 421 dlakalmkav keegfpirph yfwpllvgrr keknvqgiie ilkgmqelgv hpdqetytdy 481 vipcfdsvns arailqengc lsdsdmfsqa glrseaangn ldfvlsflks ntlpislqsi 541 rsslllgfrr smninlwsei tellykdgry cqeprgptea vgyflynlid smsdsevqak 601 eehlrqyfhq lekmnvkipe niyrgirnll esyhvpelik dahllveskn ldfqktvqlt 661 sselestlet lkaenqpird vlkqlilvlc seenmqkale lkakyesdmv tggyaalinl 721 ccrhdkveda lnlkeefdrl dssavldtgk yvglvrvlak hgklqdaini lkemkekdvl 781 ikdttalsff hmlngaalrg eietvkqlhe aivtlglaep stnisfplvt vhlekgdlst 841 alevaidcye kykvlprihd vlcklvekge tdliqkamdf vsqeqgemvm lydlffaflq 901 tgnykeakki ietpgirars arlqwfcdrc vannqvetle klveltqklf ecdrdqmyyn 961 llklykingd wqradavwnk iqeenvipre ktlrllaeil regnqevpfd vpelwyedek 1021 hslnsssast tepdfqkdil iacrlnqkkg aydiflnake qnivfnaety snlikllmse 1081 dyftqamevk afaethikgf tlndaansrl iitqvrrdyl keavttlktv ldqqqtpsrl 1141 avtrviqala mkgdveniev vqkmlngled siglskmvfi nnialaqikn nnidaaieni 1201 enmltsenkv iepqyfglay lfrkvieeql epavekisim aerlanqfai ykpvtdfflq 1261 lvdagkvdda rallqrcgai aeqtpilllf llrnsrkqgk astvksvlel ipelnekeea 1321 ynslmksyvs ekdvtsakal yehltakntk lddlflkrya sllkyagepv pfieppesfe 1381 fyaqqlrklr enss';

protein();

} else if (randomizer <= 190) {

peptidesequence = ' 1 maallrsarw llragaaprl plslrllpgg pgrlhaasyl paaragpvag gllsparlya 61 iaakekdiqe estfssrkis nqfdwalmrl dlsvrrtgri pkkllqkvfn dtcrsgglgg 121 shallllrsc gsllpelkle ertefahriw dtlqklgavy dvshynallk vylqneykfs 181 ptdflakmee aniqpnrvty qrliasycnv gdiegaskil gfmktkdlpv teavfsalvt 241 gharagdmen aeniltvmrd agiepgpdty lallnayaek gdidhvkqtl ekvekselhl 301 mdrdllqiif sfskagypqy vseilekvtc erryipdamn lilllvtekl edvalqilla 361 cpvskedgps vfgsfflqhc vtmntpvekl tdyckklkev qmhsfplqft lhcallankt 421 dlakalmkav keegfpirph yfwpllvgrr keknvqgiie ilkgmqelgv hpdqetytdy 481 vipcfdsvns arailqengc lsdsdmfsqa glrseaangn ldfvlsflks ntlpislqsi 541 rsslllgfrr smninlwsei tellykdgry cqeprgptea vgyflynlid smsdsevqak 601 eehlrqyfhq lekmnvkipe niyrgirnll esyhvpelik dahllveskn ldfqktvqlt 661 sselestlet lkaenqpird vlkqlilvlc seenmqkale lkakyesdmv tggyaalinl 721 ccrhdkveda lnlkeefdrl dssavldtgk yvglvrvlak hgklqdaini lkemkekdvl 781 ikdttalsff hmlngaalrg eietvkqlhe aivtlglaep stnisfplvt vhlekgdlst 841 alevaidcye kykvlprihd vlcklvekge tdliqkamdf vsqeqgemvm lydlffaflq 901 tgnykeakki ietpgirars arlqwfcdrc vannqvetle klveltqklf ecdrdqmyyn 961 llklykingd wqradavwnk iqeenvipre ktlrllaeil regnqevpfd vpelwyedek 1021 hslnsssast tepdfqkdil iacrlnqkkg aydiflnake qnivfnaety snlikllmse 1081 dyftqamevk afaethikgf tlndaansrl iitqvrrdyl keavttlktv ldqqqtpsrl 1141 avtrviqala mkgdveniev vqkmlngled siglskmvfi nnialaqikn nnidaaieni 1201 enmltsenkv iepqyfglay lfrkvieeql epavekisim aerlanqfai ykpvtdfflq 1261 lvdagkvdda rallqastvk svlelipeln ekeeaynslm ksyvsekdvt sakalyehlt 1321 akntklddlf lkryasllky agepvpfiep pesfefyaqq lrklrenss';

protein();

} else {

peptidesequence = '1 mllltrspta whrlsqlkpr vlpgtlggqa lhlrswllsr qgpaetggqg qpqgpglrtr 61 llitglfgag lggawlalra ekerlqqqkr tealrqaavg qgdfhlldhr grarckadfr 121 gqwvlmyfgf thcpdicpde leklvqvvrq leaepglppv qpvfitvdpe rddveamary 181 vqdfhprllg ltgstkqvaq ashsyrvyyn agpkdedqdy ivdhsiaiyl lnpdglftdy 241 ygrsrsaeqi sdsvrrhmaa frsvls';

protein();

}

}

} else {

randomizer = math\_random\_int(1, 210);

if (randomizer <= 10) {

peptidesequence = ' 1 mlsvrvaaav vralprragl vsrnalgssf iaarnfhasn thlqktgtae mssileeril 61 gadtsvdlee tgrvlsigdg iarvhglrnv qaeemvefss glkgmslnle pdnvgvvvfg 121 ndklikegdi vkrtgaivdv pvgeellgrv vdalgnaidg kgpigsktrr rvglkapgii 181 prisvrepmq tgikavdslv pigrgqreli igdrqtgkts iaidtiinqk rfndgsdekk 241 klyciyvaig qkrstvaqlv krltdadamk ytivvsatas daaplqylap ysgcsmgeyf 301 rdngkhalii yddlskqava yrqmslllrr ppgreaypgd vfylhsrlle raakmndafg 361 ggsltalpvi etqagdvsay iptnvisitd gqifletelf ykgirpainv glsvsrvgsa 421 aqtramkqva gtmklelaqy revaafaqfg sdldaatqql lsrgvrltel lkqgqyspma 481 ieeqvaviya gvrgyldkle pskitkfena flshvvsqhq allgtiradg kiseqsdakl 541 keivtnflag fea';

protein();

} else if (randomizer <= 20) {

peptidesequence = '1 mlgfvgrvaa apasgalrrl tpsaslppaq lllraaptav hpvrdyaaqt spspkagaat 61 grivavigav vdvqfdeglp pilnalevqg retrlvleva qhlgestvrt iamdgteglv 121 rgqkvldsga pikipvgpet lgrimnvige pidergpikt kqfapihaea pefmemsveq 181 eilvtgikvv dllapyakgg kiglfggagv gktvlimeli nnvakahggy svfagvgert 241 regndlyhem iesgvinlkd atskvalvyg qmneppgara rvaltgltva eyfrdqegqd 301 vllfidnifr ftqagsevsa llgripsavg yqptlatdmg tmqeritttk kgsitsvqai 361 yvpaddltdp apattfahld attvlsraia elgiypavdp ldstsrimdp nivgsehydv 421 argvqkilqd ykslqdiiai lgmdelseed kltvsrarki qrflsqpfqv aevftghmgk 481 lvplketikg fqqilageyd hlpeqafymv gpieeavaka dklaeehss';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mfsragvagl sawtlqpqwi qvrnmatlkd itrrlksikn iqkitksmkm vaaakyarae 61 relkpariyg lgslalyeka dikgpedkkk hlligvssdr glcgaihssi akqmksevat 121 ltaagkevml vgigdkirgi lyrthsdqfl vafkevgrkp ptfgdasvia lellnsgyef 181 degsiifnkf rsvisyktee kpifslntva sadsmsiydd idadvlqnyq eynlaniiyy 241 slkesttseq sarmtamdna sknasemidk ltltfnrtrq avitkeliei isgaaal';

protein();

} else if (randomizer <= 40) {

peptidesequence = ' 1 mlpaallrrp glgrlvrhar ayaeaaaapa aasgpnqmsf tfasptqvff nganvrqvdv 61 ptltgafgil aahvptlqvl rpglvvvhae dgttskyfvs sgsiavnads svqllaeeav 121 tldmldlgaa kanlekaqae lvgtadeatr aeiqiriean ealvkale';

protein();

} else if (randomizer <= 50) {

peptidesequence = ' 1 mvaywrqagl syirysqica kavrdalkte fkanaektsg snvkivkvkk e';

protein();

} else if (randomizer <= 60) {

peptidesequence = ' 1 mqtagalfis palircctrg lirpvsasfl nspvnsskqp sysnfplqva rrefqtsvvs 61 rdidtaakfi gagaatvgva gsgagigtvf gsliigyarn pslkqqlfsy ailgfalsea 121 mglfclmvaf lilfam';

protein();

} else if (randomizer <= 70) {

peptidesequence = '1 mpelilspat aphplkmfac skfvstpslv kstsqllsrp lsavvlkrpe iltdeslssl 61 avscpltslv ssrsfqtsai srdidtaakf igagaatvgv agsgagigtv fgsliigyar 121 npslkqqlfs yailgfalse amglfclmva flilfam';

protein();

} else if (randomizer <= 80) {

peptidesequence = ' 1 mfacaklact psliragsrv ayrpisasvl srpeasrtge gstvfngaqn gvsqliqref 61 qtsaisrdid taakfigaga atvgvagsga gigtvfgsli igyarnpslk qqlfsyailg 121 falseamglf clmvaflilf am';

protein();

} else if (randomizer <= 90) {

peptidesequence = '1 mvppvqvspl iklgrysalf lgvaygatry nylkpraeee rriaaeekkk qdelkriare 61 laeddsilk';

protein();

} else if (randomizer <= 100) {

peptidesequence = '1 masvgecpap vpvkdkklle vklgelpswi lmrdfspsgi fgafqrgyyr yynkyinvkk 61 gsisgitmvl acyvlfsysf sykhlkherl rkyh';

protein();

} else if (randomizer <= 110) {

peptidesequence = ' 1 maqfvrnlve ktpalvnaav tyskprlatf wyyakvelvp ptpaeiprai qslkkivnsa 61 qtgsfkqltv keavlnglva tevlmwfyvg eiigkrgiig ydv';

protein();

} else if (randomizer <= 120) {

peptidesequence = ' 1 mlqsiikniw ipmkpyytkv yqeiwigmgl mgfivykira adkrskalka sapapghh';

protein();

} else if (randomizer <= 130) {

peptidesequence = ' 1 mstamnfgtk sfqprppdkg sfpldhlgec ksfkekfmkc lhnnnfenal crkeskeyle 61 crmerklmlq epleklgfgd ltsgkseakk';

protein();

} else if (randomizer <= 140) {

peptidesequence = '1 mnenlfasfi aptilglpaa vliilfppll iptskylinn rlittqqwli kltskqmmtm 61 hntkgrtwsl mlvsliifia ttnllgllph sftpttqlsm nlamaiplwa gtvimgfrsk 121 iknalahflp qgtptplipm lviietisll iqpmalavrl tanitaghll mhligsatla 181 mstinlpstl iiftilillt ileiavaliq ayvftllvsl ylhdnt';

protein();

} else if (randomizer <= 150) {

peptidesequence = ' 1 mpqlnttvwp tmitpmlltl flitqlkmln tnyhlppspk pmkmknynkp wepkwtkics 61 lhslppqs';

protein();

} else if (randomizer <= 160) {

peptidesequence = '1 mlsrvvlsaa ataapslkna aflgpgvlqa trtfhtgqph lvpvpplpey ggkvryglip 61 eeffqflypk tgvtgpyvlg tglilyalsk eiyvisaetf talsvlgvmv ygikkygpfv 121 adfadklneq klaqleeakq asiqhiqnai dteksqqalv qkrhylfdvq rnniamalev 181 tyrerlyrvy kevknrldyh isvqnmmrrk eqehminwve khvvqsistq qeketiakci 241 adlkllakka qaqpvm';

protein();

} else if (randomizer <= 170) {

peptidesequence = '1 magrklalkt idwvafaeii pqnqkaiass lkswnetlts rlaalpenpp aidwayykan 61 vakaglvddf ekkfnalkvp vpedkytaqv daeekedvks caewvslska riveyekeme 121 kmknlipfdq mtiedlneaf petkldkkky pywphqpien l';

protein();

} else if (randomizer <= 180) {

peptidesequence = ' 1 milqrlfrfs svirsavsvh lrrnigvtav afnkeldpiq klfvdkirey kskrqtsggp 61 vdasseyqqe lerelfklkq mfgnadmntf ptfkfedpkf eviekpqa';

protein();

} else if (randomizer <= 190) {

peptidesequence = ' 1 maapavsgls rqvrcfstsv vrpfaklvrp pvqvygiegr yatalysaas kqnkleqvek 61 ellrvaqilk epkvaasvln pyvkrsikvk slnditaker fsplttnlin llaengrlsn 121 tqgvvsafst mmsvhrgevp ctvtsasple eatlselktv lksflsqgqv lkleaktdps 181 ilggmivrig ekyvdmsvkt kiqklgramr eiv';

protein();

} else if (randomizer <= 200) {

peptidesequence = '1 mavtalaart wlgvwgvrtm qargfgsdqs envdrgagsi reaggafgkr eqaeeeryfr 61 aqsreqlaal kkhheeeivh hkkeierlqk eierhkqkik mlkhdd';

protein();

} else {

peptidesequence = '1 mavtalaart wlgvwgvrtm qargfgsdqs envdrgagsi reaggafgkr eqaeeeryfr 61 aqsreqlaal kkhheeeivh hkkeierlqk eierhkqkik mlkhdd';

protein();

}

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 6.187927353e+26;

cursor1y = cursor4y - 6.187927353e+26;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(2.970205129e+28 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count11 = 0; count11 < generation\_parameter\_3; count11++) {

for (var count10 = 0; count10 < generation\_parameter\_x; count10++) {

for (var count9 = 0; count9 < generation\_parameter\_z; count9++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

coenzyme\_Q();

} else if (randomizer <= 50) {

heme\_c();

} else if (randomizer <= 75) {

cardiolipin();

} else {

peptidesequence = ' 1 mgdvekgkki fimkcsqcht vekggkhktg pnlhglfgrk tgqapgysyt aanknkgiiw 61 gedtlmeyle npkkyipgtk mifvgikkke eradliaylk katne';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 6.187927353e+26;

cursor1y = cursor4y - 2.908325856e+28;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(2.970205129e+28 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(1.23758547e+27 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count14 = 0; count14 < generation\_parameter\_3; count14++) {

for (var count13 = 0; count13 < generation\_parameter\_x; count13++) {

for (var count12 = 0; count12 < generation\_parameter\_z; count12++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 100);

if (randomizer <= 25) {

coenzyme\_Q();

} else if (randomizer <= 50) {

heme\_c();

} else if (randomizer <= 75) {

cardiolipin();

} else {

peptidesequence = ' 1 mgdvekgkki fimkcsqcht vekggkhktg pnlhglfgrk tgqapgysyt aanknkgiiw 61 gedtlmeyle npkkyipgtk mifvgikkke eradliaylk katne';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 6.187927353e+26;

cursor1y = cursor4y - 6.187927353e+26;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(1.23758547e+27 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.970205129e+28 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count17 = 0; count17 < generation\_parameter\_3; count17++) {

for (var count16 = 0; count16 < generation\_parameter\_x; count16++) {

for (var count15 = 0; count15 < generation\_parameter\_z; count15++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = ' 1 magalvrkaa dyvrskdfrd ylmsthfwgp vanwglpiaa indmkkspei isgrmtfalc 61 cysltfmrfa ykvqprnwll fachatneva qliqggrlik hemtktasa';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 marmavlwrk mrdnfqskef reyvssthfw gpafswglpl aafkdmkasp eiisgrmtta 61 lilysaifmr fayrvqprnl llmachctnv maqsvqasry llyyygggga eakardppat 121 aaaatspgsq ppkqas';

protein();

} else {

peptidesequence = ' 1 msaagarglr atyhrlldkv elmlpeklrp lynhpagprt vffwapimkw glvcagladm 61 arpaeklsta qsavlmatgf iwsryslvii pknwslfavn ffvgaagasq lfriwrynqe 121 lkakahk';

protein();

}

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 15) {

peptidesequence = ' 1 msppvysdis rnindllnkd fyhatpaafd vqtttangik fslkakqpvk dgplstnvea 61 klndkqtglg ltqgwsntnn lqtklefanl tpglknelit sltpgvaksa vlnttftepf 121 ftargafdlc lksptfvgdl tmahegivgg aefgydisag sisryamals yfakdyslga 181 tlnneqittv dffqnvnafl qvakatmmnc klpnsnvnie fatrylpdas sqvkakvsds 241 givtlaykql lrpgvtlgvg ssfdalklse pvhklgwsls fda';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mtsssgvdne isldspmpif nesstlkpir vagvvttgtd hidpsvlqay lddtimksit 61 lgqlvknadv lnkrlcqhhi alnakqsfhf qgntyisdek ethdvvplme vvsqldilpp 121 ktftaktgtn fgndndaeay lqfeklidkk ylklptrvnl eilrgtkihs sflfnsyssl 181 spqsilnlkv fsqfynwntn kgldigqrga rlslryeplf lhkllhnphs nesptlfhew 241 fletcwrstk icsqgtsapy mysgtmlsqa gdqlrtilgh tfvldkrdhi mcptkgsmlk 301 wsnelspgkh lktqlelnsv kswmnddfit fsttiktgyl knlssqqslp vhicdkfqsg 361 gpsdirgfqt fglgprdlyd avggdafvsy glsvfsrlpw kkveksnfrl hwffnggklv 421 nhdntslgnc igqlskehst stgiglvlrh pmarfelnft lpitahendl irkgfqfglg 481 lafl';

protein();

} else if (randomizer <= 45) {

peptidesequence = '1 mvssfsvpmp vkrifdtfpl qtyaaqtdkd eavaleiqrr sytfterggg sseltvegty 61 klgvynvfle antgaalatd pwclfvqlal cqknglvlpt hsqeqtpsht cnhemlvlsr 121 lsnpdealpi lvegykkrii rstvaiseim rsrilddaeq lmyytlldtv lydcwitqii 181 fcasdaqfme lyscqklsgs ivtpldvens llqklsaksl kisltkrnkf qfrhreivks 241 mqgvyhnhhn svnqeqvlnv lfenskqvll glkdmlksdg qptylhlkia syilcitnvk 301 epiklktfve neckelvqfa qdtlknfvq';

protein();

} else {

peptidesequence = ' 1 mvkgsvhlwg kdgkaslisv dsialvwfik lctseeaksm vaglqivfsn ntdlssdgkl 61 pvlildngtk vsgyvnivqf lhknictsky ekgtdyeedl aivrkkdrll eysllnyvdv 121 eisrltdyql flntknyney tkklfsklly fpmwyntplq lrsqarence eiigsltled 181 deefveskam esasqlaqsk tfkiahknki kgkqelqqvk ynlqfdnrlq scvsnwlaar 241 kklddsvils sdllflanly vqlglpdgnr irskleqtfg sellnsmsnk iddfvhrpsn 301 nleqrdpqfr eqgnvvmsly nlackyi';

protein();

}

} else {

randomizer = math\_random\_int(1, 60);

peptidesequence = ' 1 mgdhawsflk dflaggvaaa vsktavapie rvklllqvqh askqisaekq ykgiidcvvr 61 ipkeqgflsf wrgnlanvir yfptqalnfa fkdkykqlfl ggvdrhkqfw ryfagnlasg 121 gaagatslcf vypldfartr laadvgkgaa qrefhglgdc iikifksdgl rglyqgfnvs 181 vqgiiiyraa yfgvydtakg mlpdpknvhi fvswmiaqsv tavaglvsyp fdtvrrrmmm 241 qsgrkgadim ytgtvdcwrk iakdegakaf fkgawsnvlr gmggafvlvl ydeikkyv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

if (unused\_variable == 5) {

cursor1x = cursor4x - 6.187927353e+26;

cursor1y = cursor4y - 6.187927353e+26;

cursor1z = cursor4z + 1.856378206e+27;

density\_parameter\_x = 6.806720089e+25 \* 1;

density\_parameter\_y = 6.806720089e+25 \* 1;

density\_parameter\_z = 6.806720089e+25;

generation\_parameter\_x = Math.round(2.908325856e+28 / density\_parameter\_x - 0);

generation\_parameter\_3 = Math.round(2.970205129e+28 / density\_parameter\_y - 0);

generation\_parameter\_z = Math.round(5.569134618e+28 / density\_parameter\_z - 0);

for (var count20 = 0; count20 < generation\_parameter\_3; count20++) {

for (var count19 = 0; count19 < generation\_parameter\_x; count19++) {

for (var count18 = 0; count18 < generation\_parameter\_z; count18++) {

if (unused\_variable == 5) {

set\_earth\_rotation();

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 20) {

peptidesequence = ' 1 magalvrkaa dyvrskdfrd ylmsthfwgp vanwglpiaa indmkkspei isgrmtfalc 61 cysltfmrfa ykvqprnwll fachatneva qliqggrlik hemtktasa';

protein();

} else if (randomizer <= 40) {

peptidesequence = '1 marmavlwrk mrdnfqskef reyvssthfw gpafswglpl aafkdmkasp eiisgrmtta 61 lilysaifmr fayrvqprnl llmachctnv maqsvqasry llyyygggga eakardppat 121 aaaatspgsq ppkqas';

protein();

} else {

peptidesequence = ' 1 msaagarglr atyhrlldkv elmlpeklrp lynhpagprt vffwapimkw glvcagladm 61 arpaeklsta qsavlmatgf iwsryslvii pknwslfavn ffvgaagasq lfriwrynqe 121 lkakahk';

protein();

}

} else if (randomizer <= 40) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 15) {

peptidesequence = ' 1 msppvysdis rnindllnkd fyhatpaafd vqtttangik fslkakqpvk dgplstnvea 61 klndkqtglg ltqgwsntnn lqtklefanl tpglknelit sltpgvaksa vlnttftepf 121 ftargafdlc lksptfvgdl tmahegivgg aefgydisag sisryamals yfakdyslga 181 tlnneqittv dffqnvnafl qvakatmmnc klpnsnvnie fatrylpdas sqvkakvsds 241 givtlaykql lrpgvtlgvg ssfdalklse pvhklgwsls fda';

protein();

} else if (randomizer <= 30) {

peptidesequence = ' 1 mtsssgvdne isldspmpif nesstlkpir vagvvttgtd hidpsvlqay lddtimksit 61 lgqlvknadv lnkrlcqhhi alnakqsfhf qgntyisdek ethdvvplme vvsqldilpp 121 ktftaktgtn fgndndaeay lqfeklidkk ylklptrvnl eilrgtkihs sflfnsyssl 181 spqsilnlkv fsqfynwntn kgldigqrga rlslryeplf lhkllhnphs nesptlfhew 241 fletcwrstk icsqgtsapy mysgtmlsqa gdqlrtilgh tfvldkrdhi mcptkgsmlk 301 wsnelspgkh lktqlelnsv kswmnddfit fsttiktgyl knlssqqslp vhicdkfqsg 361 gpsdirgfqt fglgprdlyd avggdafvsy glsvfsrlpw kkveksnfrl hwffnggklv 421 nhdntslgnc igqlskehst stgiglvlrh pmarfelnft lpitahendl irkgfqfglg 481 lafl';

protein();

} else if (randomizer <= 45) {

peptidesequence = '1 mvssfsvpmp vkrifdtfpl qtyaaqtdkd eavaleiqrr sytfterggg sseltvegty 61 klgvynvfle antgaalatd pwclfvqlal cqknglvlpt hsqeqtpsht cnhemlvlsr 121 lsnpdealpi lvegykkrii rstvaiseim rsrilddaeq lmyytlldtv lydcwitqii 181 fcasdaqfme lyscqklsgs ivtpldvens llqklsaksl kisltkrnkf qfrhreivks 241 mqgvyhnhhn svnqeqvlnv lfenskqvll glkdmlksdg qptylhlkia syilcitnvk 301 epiklktfve neckelvqfa qdtlknfvq';

protein();

} else {

peptidesequence = ' 1 mvkgsvhlwg kdgkaslisv dsialvwfik lctseeaksm vaglqivfsn ntdlssdgkl 61 pvlildngtk vsgyvnivqf lhknictsky ekgtdyeedl aivrkkdrll eysllnyvdv 121 eisrltdyql flntknyney tkklfsklly fpmwyntplq lrsqarence eiigsltled 181 deefveskam esasqlaqsk tfkiahknki kgkqelqqvk ynlqfdnrlq scvsnwlaar 241 kklddsvils sdllflanly vqlglpdgnr irskleqtfg sellnsmsnk iddfvhrpsn 301 nleqrdpqfr eqgnvvmsly nlackyi';

protein();

}

} else {

randomizer = math\_random\_int(1, 60);

peptidesequence = ' 1 mgdhawsflk dflaggvaaa vsktavapie rvklllqvqh askqisaekq ykgiidcvvr 61 ipkeqgflsf wrgnlanvir yfptqalnfa fkdkykqlfl ggvdrhkqfw ryfagnlasg 121 gaagatslcf vypldfartr laadvgkgaa qrefhglgdc iikifksdgl rglyqgfnvs 181 vqgiiiyraa yfgvydtakg mlpdpknvhi fvswmiaqsv tavaglvsyp fdtvrrrmmm 241 qsgrkgadim ytgtvdcwrk iakdegakaf fkgawsnvlr gmggafvlvl ydeikkyv';

protein();

}

}

cursor1z = cursor1z + density\_parameter\_z;

}

cursor1z = cursor1z - generation\_parameter\_z \* density\_parameter\_z;

cursor1x = cursor1x - density\_parameter\_x;

}

cursor1x = cursor1x + density\_parameter\_x \* generation\_parameter\_x;

cursor1y = cursor1y - density\_parameter\_y;

}

}

cursor1x = cursor4x;

cursor1y = cursor4y;

cursor1z = cursor4z;

}

function protein() {

cursor3x = cursor1x;

cursor3y = cursor1y;

cursor3z = cursor1z;

refine\_protein\_sequence();

make = true;

loc1 = [cursor3z, cursor3y, cursor3x];

loc2 = [cursor3z + (1.546981838e+25 + genderiv.length \* 2.165774573e+25), cursor3y - 6.744840815e+25, cursor3x - 3.087775749e+26];

screen\_molecules();

if (make == true) {

protein\_cap\_1();

intextnum = 1;

while (intextnum <= genderiv.length) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

alanine();

} else if (genderiv.charAt(intextnum - 1) == 'r' || genderiv.charAt(intextnum - 1) == 'R') {

arginine();

} else if (genderiv.charAt(intextnum - 1) == 'n' || genderiv.charAt(intextnum - 1) == 'N') {

aspargine();

} else if (genderiv.charAt(intextnum - 1) == 'd' || genderiv.charAt(intextnum - 1) == 'D') {

aspartic\_acid();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cysteine();

} else if (genderiv.charAt(intextnum - 1) == 'e' || genderiv.charAt(intextnum - 1) == 'E') {

glutamic\_acid();

} else if (genderiv.charAt(intextnum - 1) == 'q' || genderiv.charAt(intextnum - 1) == 'Q') {

glutamine();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

glycine();

} else if (genderiv.charAt(intextnum - 1) == 'h' || genderiv.charAt(intextnum - 1) == 'H') {

histidine();

} else if (genderiv.charAt(intextnum - 1) == 'i' || genderiv.charAt(intextnum - 1) == 'I') {

isoleucine();

} else if (genderiv.charAt(intextnum - 1) == 'l' || genderiv.charAt(intextnum - 1) == 'L') {

leucine();

} else if (genderiv.charAt(intextnum - 1) == 'k' || genderiv.charAt(intextnum - 1) == 'K') {

lysine();

} else if (genderiv.charAt(intextnum - 1) == 'm' || genderiv.charAt(intextnum - 1) == 'M') {

methionine();

} else if (genderiv.charAt(intextnum - 1) == 'f' || genderiv.charAt(intextnum - 1) == 'F') {

phenylalanine();

} else if (genderiv.charAt(intextnum - 1) == 'p' || genderiv.charAt(intextnum - 1) == 'P') {

proline();

} else if (genderiv.charAt(intextnum - 1) == 's' || genderiv.charAt(intextnum - 1) == 'S') {

serine();

} else if (genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') {

threorine();

} else if (genderiv.charAt(intextnum - 1) == 'w' || genderiv.charAt(intextnum - 1) == 'W') {

tryptophan();

} else if (genderiv.charAt(intextnum - 1) == 'y' || genderiv.charAt(intextnum - 1) == 'Y') {

tyrosine();

} else if (genderiv.charAt(intextnum - 1) == 'v' || genderiv.charAt(intextnum - 1) == 'V') {

valine();

}

intextnum = intextnum + 1;

}

protein\_cap\_2();

}

cursor1x = cursor3x;

cursor1y = cursor3y;

cursor1z = cursor3z;

}

function rna() {

cursor3x = cursor1x;

cursor3y = cursor1y;

cursor3z = cursor1z;

refine\_rna\_sequence();

make = true;

loc1 = [cursor3z, cursor3y, cursor3x];

loc2 = [cursor3z + (1.546981838e+25 + genderiv.length \* 2.1038953e+25), cursor3y - 6.187927353e+25, cursor3x - 2.1038953e+25];

screen\_molecules();

if (make == true) {

rna\_cap\_1();

intextnum = 1;

while (intextnum <= genderiv.length) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

rna\_adenine();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

rna\_guanine();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

rna\_cytosine();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

rna\_uracil();

}

intextnum = intextnum + 1;

}

rna\_cap\_2();

}

cursor1x = cursor3x;

cursor1y = cursor3y;

cursor1z = cursor3z;

}

function refine\_dna\_sequence() {

gentarget = genome;

get\_genetic\_text();

genderiv = '';

intextnum = 1;

while (intextnum <= genraw.length) {

if (genraw.charAt(intextnum - 1) == 'a' || genraw.charAt(intextnum - 1) == 'A') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'g' || genraw.charAt(intextnum - 1) == 'G') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'c' || genraw.charAt(intextnum - 1) == 'C') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if ((genraw.charAt(intextnum - 1) == 't' || genraw.charAt(intextnum - 1) == 'T') || (genraw.charAt(intextnum - 1) == 'u' || genraw.charAt(intextnum - 1) == 'U')) {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

}

intextnum = intextnum + 1;

}

var repeat\_end = math\_random\_int(1670, 2000);

for (var count = 0; count < repeat\_end; count++) {

genderiv = String('TTAGGG') + String(genderiv);

}

var repeat\_end2 = math\_random\_int(1670, 2000);

for (var count2 = 0; count2 < repeat\_end2; count2++) {

genderiv = String(genderiv) + String('TTAGGG');

}

}

function refine\_protein\_sequence() {

genraw = peptidesequence;

aaseq = '';

intextnum = 1;

while (intextnum <= genraw.length) {

if (genraw.charAt(intextnum - 1) == 'a' || genraw.charAt(intextnum - 1) == 'A') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'r' || genraw.charAt(intextnum - 1) == 'R') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'n' || genraw.charAt(intextnum - 1) == 'N') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'd' || genraw.charAt(intextnum - 1) == 'D') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'c' || genraw.charAt(intextnum - 1) == 'C') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'e' || genraw.charAt(intextnum - 1) == 'E') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'q' || genraw.charAt(intextnum - 1) == 'Q') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'g' || genraw.charAt(intextnum - 1) == 'G') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'h' || genraw.charAt(intextnum - 1) == 'H') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'i' || genraw.charAt(intextnum - 1) == 'I') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'l' || genraw.charAt(intextnum - 1) == 'L') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'k' || genraw.charAt(intextnum - 1) == 'K') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'm' || genraw.charAt(intextnum - 1) == 'M') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'f' || genraw.charAt(intextnum - 1) == 'F') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'p' || genraw.charAt(intextnum - 1) == 'P') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 's' || genraw.charAt(intextnum - 1) == 'S') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 't' || genraw.charAt(intextnum - 1) == 'T') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'w' || genraw.charAt(intextnum - 1) == 'W') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'y' || genraw.charAt(intextnum - 1) == 'Y') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'v' || genraw.charAt(intextnum - 1) == 'V') {

aaseq = String(aaseq) + String(genraw.charAt(intextnum - 1));

}

intextnum = intextnum + 1;

}

}

function refine\_rna\_sequence() {

genraw = rnaseq;

genderiv = '';

intextnum = 1;

while (intextnum <= genraw.length) {

if (genraw.charAt(intextnum - 1) == 'a' || genraw.charAt(intextnum - 1) == 'A') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'g' || genraw.charAt(intextnum - 1) == 'G') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if (genraw.charAt(intextnum - 1) == 'c' || genraw.charAt(intextnum - 1) == 'C') {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

} else if ((genraw.charAt(intextnum - 1) == 't' || genraw.charAt(intextnum - 1) == 'T') || (genraw.charAt(intextnum - 1) == 'u' || genraw.charAt(intextnum - 1) == 'U')) {

genderiv = String(genderiv) + String(genraw.charAt(intextnum - 1));

}

intextnum = intextnum + 1;

}

}

function nucleoid() {

cursor3x = cursor1x;

cursor3y = cursor1y;

cursor3z = cursor1z;

set\_earth\_rotation();

dna = true;

make = true;

ploidynum = 1;

refine\_dna\_sequence();

cursor1x = cursor3x;

cursor1y = cursor3y;

cursor1z = cursor3z + 2.1038953e+25 \* 8;

if (make == true) {

dna\_cap\_1();

cursor1z = cursor1z + 2.1038953e+25 \* 1;

intextnum = 1;

while (dna == true) {

while (dna == true && cursor1y > cursor3y - (Math.floor((nucleoid\_size\_y - 2.1038953e+25 \* 8) / (2.1038953e+25 \* 16)) - 1) \* 3.36623248e+26) {

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 4) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_forth\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_forth\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

cursor1y = cursor1y - 2.1038953e+25 \* 4;

for (var count = 0; count < 8; count++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_up\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_up\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_up\_forth();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_up\_forth();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y - 2.1038953e+25 \* 1;

}

}

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 8) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_back\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_back\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

cursor1z = cursor1z - 2.1038953e+25 \* 4;

cursor1y = cursor1y - 2.1038953e+25 \* 1;

for (var count2 = 0; count2 < 6; count2++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_up\_back();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_up\_back();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_up\_back();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_up\_back();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y - 2.1038953e+25 \* 1;

}

}

cursor1z = cursor1z + 2.1038953e+25 \* 4;

cursor1y = cursor1y + 2.1038953e+25 \* 3;

}

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 8) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_forth\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_forth\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

cursor1x = cursor1x - 2.1038953e+25 \* 1;

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_left\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_left\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_left\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_left\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

}

intextnum = intextnum + 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 8) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_back\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_back\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

while (dna == true && cursor1y < cursor3y) {

cursor1z = cursor1z - 2.1038953e+25 \* 4;

cursor1y = cursor1y - 2.1038953e+25 \* 2;

for (var count3 = 0; count3 < 6; count3++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_down\_back();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_down\_back();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_down\_back();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_down\_back();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y + 2.1038953e+25 \* 1;

}

}

cursor1z = cursor1z + 2.1038953e+25 \* 4;

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 4) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_forth\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_forth\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

for (var count4 = 0; count4 < 8; count4++) {

if (dna == true) {

cursor1y = cursor1y + 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_down\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_down\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_down\_forth();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_down\_forth();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

}

cursor1y = cursor1y + 2.1038953e+25 \* 4;

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 8) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_back\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_back\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

}

cursor1z = cursor1z - 2.1038953e+25 \* 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_left\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

guanine\_left\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

cytosine\_left\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_left\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

}

intextnum = intextnum + 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

dna\_cap\_2();

dna = true;

cursor1x = cursor3x;

cursor1y = cursor3y - 2.1038953e+25 \* 4;

cursor1z = cursor3z + 2.1038953e+25 \* 8;

dna\_cap\_2();

cursor1z = cursor1z + 2.1038953e+25 \* 1;

intextnum = 1;

while (dna == true) {

while (dna == true && cursor1y > (cursor3y - 2.1038953e+25 \* 4) - (Math.floor((nucleoid\_size\_y - 2.1038953e+25 \* 4) / (2.1038953e+25 \* 16)) - 1) \* 3.36623248e+26) {

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 8) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_back\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_back\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

cursor1y = cursor1y - 2.1038953e+25 \* 1;

for (var count5 = 0; count5 < 6; count5++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_down\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_down\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_down\_forth();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_down\_forth();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y - 2.1038953e+25 \* 1;

}

}

cursor1y = cursor1y + 2.1038953e+25 \* 3;

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 4) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_forth\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_forth\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_forth\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

cursor1y = cursor1y - 2.1038953e+25 \* 4;

cursor1z = cursor1z - 2.1038953e+25 \* 4;

for (var count6 = 0; count6 < 8; count6++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_down\_back();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_down\_back();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_down\_back();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_down\_back();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y - 2.1038953e+25 \* 1;

}

}

cursor1z = cursor1z + 2.1038953e+25 \* 4;

}

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 8) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_back\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_back\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_back\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

cursor1x = cursor1x - 2.1038953e+25 \* 1;

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_left\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_left\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_left\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_left\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

}

intextnum = intextnum + 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 8) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_forth\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_forth\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

while (dna == true && cursor1y < cursor3y - 2.1038953e+25 \* 4) {

for (var count7 = 0; count7 < 4; count7++) {

if (dna == true) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_forth\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_forth\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

}

cursor1z = cursor1z - 2.1038953e+25 \* 4;

cursor1y = cursor1y + 2.1038953e+25 \* 1;

for (var count8 = 0; count8 < 8; count8++) {

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_up\_back();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_up\_back();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_up\_back();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_up\_back();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1y = cursor1y + 2.1038953e+25 \* 1;

}

}

cursor1y = cursor1y + 2.1038953e+25 \* 3;

cursor1z = cursor1z + 2.1038953e+25 \* 4;

while (dna == true && cursor1z < cursor3z + Math.floor(nucleoid\_size\_z / 2.1038953e+25 - 8) \* 2.1038953e+25) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_back\_down();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_back\_down();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_back\_down();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

cursor1y = cursor1y - 2.1038953e+25 \* 3;

for (var count9 = 0; count9 < 6; count9++) {

if (dna == true) {

cursor1y = cursor1y + 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_up\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_up\_forth();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_up\_forth();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_up\_forth();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

}

cursor1y = cursor1y + 2.1038953e+25 \* 1;

while (dna == true && cursor1z > cursor3z + 2.1038953e+25 \* 8) {

cursor1z = cursor1z - 2.1038953e+25 \* 1;

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

thymine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_forth\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_forth\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

adenine\_forth\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

intextnum = intextnum + 1;

}

}

cursor1z = cursor1z - 2.1038953e+25 \* 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

if (dna == true) {

if (genderiv.charAt(intextnum - 1) == 'a' || genderiv.charAt(intextnum - 1) == 'A') {

adenine\_left\_up();

} else if (genderiv.charAt(intextnum - 1) == 'g' || genderiv.charAt(intextnum - 1) == 'G') {

cytosine\_left\_up();

} else if (genderiv.charAt(intextnum - 1) == 'c' || genderiv.charAt(intextnum - 1) == 'C') {

guanine\_left\_up();

} else if ((genderiv.charAt(intextnum - 1) == 't' || genderiv.charAt(intextnum - 1) == 'T') || (genderiv.charAt(intextnum - 1) == 'u' || genderiv.charAt(intextnum - 1) == 'U')) {

thymine\_left\_up();

}

if (intextnum >= genderiv.length) {

if (ploidynum < ploidy) {

ploidynum = ploidynum + 1;

intextnum = 1;

} else {

dna = false;

}

}

}

intextnum = intextnum + 1;

cursor1x = cursor1x - 2.1038953e+25 \* 1;

cursor1z = cursor1z + 2.1038953e+25 \* 1;

}

dna\_cap\_1();

}

cursor1x = cursor3x;

cursor1y = cursor3y;

cursor1z = cursor3z;

}

function air\_hydrogen() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_nitrogen() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

nitrogen\_back\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

nitrogen\_forth\_3();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_oxygen() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function bicarbonate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922e+25;

molsizey = 1.922e+25;

molsizez = 1.922e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

graphene\_anion();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function carbon\_dioxide() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_62\_carbon\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_forth\_2();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ferrosulphur() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922e+25;

molsizey = 1.922e+25;

molsizez = 1.922e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

iron\_horizontal();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

iron\_horizontal();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

my\_63\_sulphur();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ozone() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_ozone\_middle();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_forth\_2();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function water\_molecule() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function iron\_sulphur\_cluster() {

ferrosulphur();

list\_molecules[list\_molecules.length - 1] = ['ferrosulphur', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

function SO4\_anion() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922068e+25;

molsizey = 1.922068e+25;

molsizez = 1.922068e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

sulphate();

list\_molecules[list\_molecules.length - 1] = ['sulfate', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ascorbic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.187927353e+25;

molsizey = 1.546981838e+25;

molsizez = 6.187927353e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.435 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.33 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['ascorbic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function auxin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.181739426e+25;

molsizey = 1.546981838e+25;

molsizez = 6.181739426e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.44 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.44 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.44 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.52 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['auxin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function biotin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.187927353e+25;

molsizey = 1.856378206e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.47 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.53 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.53 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.01 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.79 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.735 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['biotin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function borate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922068e+25;

molsizey = 1.922068e+25;

molsizez = 1.922068e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

borophene();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_anion\_back();

}

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cellulose() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 5.086476e+25;

molsizey = 4.826583e+25;

molsizez = 6.435444e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.325 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.535 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.255 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.535 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.53 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.765 \* molsizez) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.675 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.475 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.49 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.725 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.725 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.725 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.725 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.825 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.325 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.535 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.255 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.535 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.53 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.425 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.765 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['cellulose', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function glucose() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.181739426e+25;

molsizey = 4.331549147e+25;

molsizez = 6.181739426e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.01 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['glucose', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function lipoic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 1.856378206e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.88 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.01 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['lipoic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function molybdate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922068e+25;

molsizey = 1.922068e+25;

molsizez = 1.922068e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

tetramolybdenum();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

oxygen\_forth\_2();

}

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function niacin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.331549147e+25;

molsizey = 1.856378206e+25;

molsizez = 4.331549147e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['niacin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function sulphate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922068e+25;

molsizey = 1.922068e+25;

molsizez = 1.922068e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

tetrasulphur();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

oxygen\_forth\_2();

}

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function alanine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['alanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function cytokinin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.181739426e+25;

molsizey = 1.856378206e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

heteronitrogen\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

heteronitrogen\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.36 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.36 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.46 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.05 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.05 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.46 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_down();

}

list\_molecules[list\_molecules.length - 1] = ['cytokinin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function default\_amino\_acid() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

oxygen\_forth\_2();

}

}

function glycine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['glycine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function protein\_cap\_1() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+26;

molsizey = 6.744840815e+25;

molsizez = 7.734909192e+24;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function protein\_cap\_2() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+26;

molsizey = 6.744840815e+25;

molsizez = 7.734909192e+24;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function aspartic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['aspartic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function cysteine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cysteine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function glutamate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

oxygen\_anion\_forth();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

oxygen\_forth\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_anion\_forth();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['glutamate', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function glutamic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['glutamic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function selenocysteine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+26;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_amino\_acid();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_11\_selenium();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['selenocysteine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function serine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['serine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function threorine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['threorine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function valine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['valine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function isoleucine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['isoleucine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function leucine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.925 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.335 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['leucine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function lysine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.52 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['lysine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function methionine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['methionine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function phenylalanine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.175 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['phenylalanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function proline() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.13 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

oxygen\_forth\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.33 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.33 \* molsizey) + 0;

cursor1z = (anchorz + 0.375 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['proline', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function tyrosine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.175 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.056 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['tyrosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function arginine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.535 \* molsizex) + 0;

cursor1y = (anchory - 0.435 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.735 \* molsizex) + 0;

cursor1y = (anchory - 0.435 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.875 \* molsizex) + 0;

cursor1y = (anchory - 0.435 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.875 \* molsizex) + 0;

cursor1y = (anchory - 0.435 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.445 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.445 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.445 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.445 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['arginine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function aspargine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.335 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['asparginine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function glutamine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['glutamine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function histidine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.175 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

heteronitrogen\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['histidine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function tryptophan() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.032084403e+25;

molsizey = 6.744840815e+25;

molsizez = 2.165774573e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

default\_amino\_acid();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.175 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.22 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['histidine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function acetylcholine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 6.187927353e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['acetylcholine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function alpha\_linoleic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.640945515e+25;

molsizey = 4.640945515e+25;

molsizez = 1.546981838e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.03 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.06 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.02 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count = 0; count < 7; count++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count2 = 0; count2 < 3; count2++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count3 = 0; count3 < 1; count3++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count4 = 0; count4 < 7; count4++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count5 = 0; count5 < 3; count5++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count6 = 0; count6 < 1; count6++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count7 = 0; count7 < 7; count7++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count8 = 0; count8 < 3; count8++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count9 = 0; count9 < 1; count9++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count10 = 0; count10 < 7; count10++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count11 = 0; count11 < 3; count11++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count12 = 0; count12 < 1; count12++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count13 = 0; count13 < 7; count13++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count14 = 0; count14 < 3; count14++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count15 = 0; count15 < 1; count15++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count16 = 0; count16 < 7; count16++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count17 = 0; count17 < 3; count17++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count18 = 0; count18 < 1; count18++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count19 = 0; count19 < 7; count19++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count20 = 0; count20 < 3; count20++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count21 = 0; count21 < 1; count21++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count22 = 0; count22 < 7; count22++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count23 = 0; count23 < 3; count23++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count24 = 0; count24 < 1; count24++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count25 = 0; count25 < 7; count25++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count26 = 0; count26 < 3; count26++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count27 = 0; count27 < 1; count27++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

list\_molecules[list\_molecules.length - 1] = ['alpha linoleic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function arachidonic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 1.23758547e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.015 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.28 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

allographene\_double\_5();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.64 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

allographene\_double\_5();

}

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.657 \* molsizey) + 0;

cursor1z = (anchorz + 0.956 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.28 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.64 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['arachidonic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function linoleic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.640945515e+25;

molsizey = 4.640945515e+25;

molsizez = 1.546981838e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.03 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.06 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.02 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.04 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count28 = 0; count28 < 7; count28++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count29 = 0; count29 < 2; count29++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count30 = 0; count30 < 4; count30++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count31 = 0; count31 < 7; count31++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count32 = 0; count32 < 2; count32++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count33 = 0; count33 < 4; count33++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count34 = 0; count34 < 7; count34++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count35 = 0; count35 < 2; count35++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count36 = 0; count36 < 4; count36++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count37 = 0; count37 < 7; count37++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count38 = 0; count38 < 2; count38++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count39 = 0; count39 < 4; count39++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count40 = 0; count40 < 7; count40++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count41 = 0; count41 < 2; count41++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count42 = 0; count42 < 4; count42++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

for (var count43 = 0; count43 < 7; count43++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count44 = 0; count44 < 2; count44++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count45 = 0; count45 < 4; count45++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count46 = 0; count46 < 7; count46++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count47 = 0; count47 < 2; count47++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count48 = 0; count48 < 4; count48++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count49 = 0; count49 < 7; count49++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count50 = 0; count50 < 2; count50++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count51 = 0; count51 < 4; count51++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

for (var count52 = 0; count52 < 7; count52++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count53 = 0; count53 < 2; count53++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

for (var count54 = 0; count54 < 4; count54++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

list\_molecules[list\_molecules.length - 1] = ['linoleic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function quinoline\_yellow() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 6.187927353e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.475 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.475 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.475 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.72 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.265 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.73 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.73 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.475 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.72 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.43 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.43 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['quinoline yellow', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

//

function randomize\_lipid\_1() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 1.5469818e+26;

molsizez = 4.0859067e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.97 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.985 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.985 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.985 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.97 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.96 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.97 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.98 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.97 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.97 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.935 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.935 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.995 \* molsizez) + 0;

for (var count = 0; count < 15; count++) {

tetracarbon();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.993 \* molsizez) + 0;

for (var count2 = 0; count2 < 15; count2++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.993 \* molsizez) + 0;

for (var count3 = 0; count3 < 15; count3++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.975 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count4 = 0; count4 < 7; count4++) {

tetracarbon();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

for (var count5 = 0; count5 < 1; count5++) {

carbon\_double\_1();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

carbon\_double\_2();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

tetracarbon();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

for (var count6 = 0; count6 < 7; count6++) {

tetracarbon();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

for (var count7 = 0; count7 < 7; count7++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

for (var count8 = 0; count8 < 1; count8++) {

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

for (var count9 = 0; count9 < 7; count9++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

for (var count10 = 0; count10 < 15; count10++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.03 \* molsizey) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.02 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.525 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.525 \* molsizey) + 0;

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function randomize\_lipid\_2() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 1.5469818e+26;

molsizez = 4.0859067e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.05 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.05 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.17 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.02 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.185 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.185 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.185 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.17 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.16 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.17 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.18 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.17 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.17 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.995 \* molsizez) + 0;

for (var count11 = 0; count11 < 15; count11++) {

tetracarbon();

cursor1y = (cursor1y - 0.04 \* molsizey) + 0;

}

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.993 \* molsizez) + 0;

for (var count12 = 0; count12 < 15; count12++) {

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.04 \* molsizey) + 0;

}

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.993 \* molsizez) + 0;

for (var count13 = 0; count13 < 15; count13++) {

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.04 \* molsizey) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.975 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count14 = 0; count14 < 7; count14++) {

tetracarbon();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

for (var count15 = 0; count15 < 1; count15++) {

carbon\_double\_2();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

carbon\_double\_1();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

tetracarbon();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

for (var count16 = 0; count16 < 7; count16++) {

tetracarbon();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

hydrogen\_up();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

for (var count17 = 0; count17 < 7; count17++) {

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

for (var count18 = 0; count18 < 1; count18++) {

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

for (var count19 = 0; count19 < 7; count19++) {

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

for (var count20 = 0; count20 < 15; count20++) {

my\_69\_hydrogen();

cursor1y = (cursor1y - 0.03 \* molsizey) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.02 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.525 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.525 \* molsizey) + 0;

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.5875 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function randomize\_lipid\_3() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 4.0859067e+25;

molsizez = 1.5469818e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.0875 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1625 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.0375 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.0625 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.0375 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.0625 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.025 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.0125 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.0375 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.0875 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.0875 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1925 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2125 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1825 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1825 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2225 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2425 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2225 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2425 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2225 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2425 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

for (var count = 0; count < 15; count++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

for (var count2 = 0; count2 < 15; count2++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

for (var count3 = 0; count3 < 15; count3++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2475 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

for (var count4 = 0; count4 < 7; count4++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count5 = 0; count5 < 1; count5++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count6 = 0; count6 < 7; count6++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

for (var count7 = 0; count7 < 7; count7++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count8 = 0; count8 < 1; count8++) {

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count9 = 0; count9 < 7; count9++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

for (var count10 = 0; count10 < 7; count10++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count11 = 0; count11 < 1; count11++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

for (var count12 = 0; count12 < 7; count12++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.02 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function randomize\_lipid\_4() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 4.0859067e+25;

molsizez = 1.5469818e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.725 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7875 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.825 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8625 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.7375 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.7625 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.7375 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.7625 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.725 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.7125 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.7375 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.7875 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.7875 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.825 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.725 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8925 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9125 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8825 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8825 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9225 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9425 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.9225 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.9425 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.9225 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.9425 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.985 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.975 \* molsizez) + 0;

for (var count13 = 0; count13 < 15; count13++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

my\_69\_hydrogen();

cursor1x = (anchorx - 0.925 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

for (var count14 = 0; count14 < 14; count14++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.05 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.34 \* molsizey) + 0;

cursor1z = (anchorz + 0.975 \* molsizez) + 0;

for (var count15 = 0; count15 < 15; count15++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9475 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.995 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count16 = 0; count16 < 7; count16++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count17 = 0; count17 < 1; count17++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count18 = 0; count18 < 7; count18++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

my\_69\_hydrogen();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

for (var count19 = 0; count19 < 7; count19++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count20 = 0; count20 < 1; count20++) {

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count21 = 0; count21 < 7; count21++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count22 = 0; count22 < 7; count22++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count23 = 0; count23 < 1; count23++) {

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

cursor1x = (cursor1x + 0.05 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.05 \* molsizex) + 0;

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

for (var count24 = 0; count24 < 7; count24++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.05 \* molsizez) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.84 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.42 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function randomize\_lipid\_5() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.5469818e+26;

molsizey = 4.0859067e+25;

molsizez = 4.0859067e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.925 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.975 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.975 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.875 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.825 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.825 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.925 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.99375 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.875 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.9365 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.93125 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2225 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8375 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.8375 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.8625 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.8385 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.997 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.8375 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count = 0; count < 15; count++) {

tetracarbon();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

cursor1y = (anchory - 0.35 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.8075 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count2 = 0; count2 < 14; count2++) {

hydrogen\_up();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

cursor1x = (anchorx - 0.8375 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count3 = 0; count3 < 15; count3++) {

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.9525 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.9525 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.985 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.9875 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.985 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.9535 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.9525 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count4 = 0; count4 < 7; count4++) {

tetracarbon();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count5 = 0; count5 < 1; count5++) {

allographene\_double\_3();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

tetracarbon();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count6 = 0; count6 < 7; count6++) {

tetracarbon();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.9225 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count7 = 0; count7 < 6; count7++) {

hydrogen\_up();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count8 = 0; count8 < 1; count8++) {

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count9 = 0; count9 < 7; count9++) {

hydrogen\_up();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

cursor1x = (anchorx - 0.9525 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count10 = 0; count10 < 7; count10++) {

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count11 = 0; count11 < 1; count11++) {

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

cursor1z = (cursor1z - 0.06 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

for (var count12 = 0; count12 < 7; count12++) {

hydrogen\_down();

cursor1x = (cursor1x + 0.03 \* molsizex) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.98 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.95 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.92 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.92 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.58 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.84 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.42 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function randomize\_lipid\_6() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.5469818e+26;

molsizey = 4.0859067e+25;

molsizez = 4.0859067e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 70) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.05 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.125 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.175 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.175 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.075 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.025 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.025 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.125 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.19375 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.075 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.1365 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.13125 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2225 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.0375 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.0375 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.0625 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.0375 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.997 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.0375 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count13 = 0; count13 < 15; count13++) {

tetracarbon();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

cursor1y = (anchory - 0.35 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.0675 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count14 = 0; count14 < 14; count14++) {

hydrogen\_up();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

cursor1x = (anchorx - 0.0375 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.999 \* molsizez) + 0;

for (var count15 = 0; count15 < 15; count15++) {

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1525 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.98 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.1525 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.985 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.1875 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.985 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.1425 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.1525 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count16 = 0; count16 < 7; count16++) {

tetracarbon();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count17 = 0; count17 < 1; count17++) {

allographene\_double\_2();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

tetracarbon();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count18 = 0; count18 < 7; count18++) {

tetracarbon();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

cursor1z = (anchorz + 0.987 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.1825 \* molsizex) + 0;

cursor1y = (anchory - 0.37 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count19 = 0; count19 < 6; count19++) {

hydrogen\_up();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count20 = 0; count20 < 1; count20++) {

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count21 = 0; count21 < 7; count21++) {

hydrogen\_up();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

cursor1x = (anchorx - 0.1525 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

for (var count22 = 0; count22 < 7; count22++) {

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count23 = 0; count23 < 1; count23++) {

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

cursor1z = (cursor1z - 0.06 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

cursor1z = (cursor1z + 0.03 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

for (var count24 = 0; count24 < 7; count24++) {

hydrogen\_down();

cursor1x = (cursor1x - 0.03 \* molsizex) + 0;

}

}

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.02 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.05 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.08 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.33 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

carbon\_pyramid\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.08 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.37 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.42 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.33 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1x = (anchorx - 0.33 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.16 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.58 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

hydrogen\_up();

}

}

list\_molecules[list\_molecules.length - 1] = ['lipid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ADP() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['ADP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function AMP() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['AMP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ATP() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['ATP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function coenzyme\_A() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 1.68311624e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.534 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.534 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.575 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.575 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.575 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.555 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.625 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.675 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.6975 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.459 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.24 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.575 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.675 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.6965 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.575 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.595 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.567 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.675 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.675 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.635 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.725 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.725 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.775 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.835 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.76 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.76 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.7495 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7705 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.225 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.175 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.075 \* molsizez) + 0;

my\_63\_sulphur();

cursor1z = (anchorz + 0.055 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1y = (anchory - 0.6135 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6135 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.225 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.225 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['coenzyme A', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function coenzyme\_Q() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.331549147e+25;

molsizey = 4.331549147e+25;

molsizez = 3.093963676e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.045 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.035 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.055 \* molsizez) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.045 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.035 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.055 \* molsizez) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.095 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.085 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.105 \* molsizez) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.08 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.075 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

for (var count = 0; count < 10; count++) {

carbon\_pyramid\_1();

cursor1z = (cursor1z + 0.02 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (cursor1z + 0.02 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (cursor1z + 0.02 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.06 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

hydrogen\_down();

cursor1z = (cursor1z + 0.02 \* molsizez) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (cursor1z - 0.02 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (cursor1z - 0.01 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (cursor1z + 0.02 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (cursor1z - 0.01 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['coenzyme Q', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_mononucleotide() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['GTP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_forth\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_1();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_mononucleotide() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['CTP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function rna\_cytosine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

default\_rna();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function rna\_uracil() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

default\_rna();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['uracyl', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function thymine\_forth\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_1();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function uracil\_mononucleotide() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.14 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.27 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.24 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['UTP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function NAD() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25 \* 2;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.39 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.125 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.46 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.825 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.625 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allonitrophene\_cation();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.565 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.565 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.92 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['NAD', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function NADH() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25 \* 2;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.39 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.125 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.46 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.825 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.625 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allonitrophene\_cation();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.565 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.565 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.92 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.835 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['NADH', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function FAD() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25 \* 2;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.39 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.125 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.46 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.825 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.625 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

heteronitrogen\_1();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_3();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.515 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.515 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.457 \* molsizex) + 0;

cursor1y = (anchory - 0.83 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['FAD', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function FADH() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25 \* 2;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.39 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.125 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.135 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_vertical();

if (unused\_variable == 5) {

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.46 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.34 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

oxygen\_back\_2();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.365 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.825 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.625 \* molsizey) + 0;

cursor1z = (anchorz + 0.36 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.645 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.357 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.643 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_1();

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.515 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.515 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.457 \* molsizex) + 0;

cursor1y = (anchory - 0.83 \* molsizey) + 0;

cursor1z = (anchorz + 0.54 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

cursor1y = (anchory - 0.835 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.47 \* molsizex) + 0;

cursor1y = (anchory - 0.835 \* molsizey) + 0;

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['FADH', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_forth\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_1();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function default\_dna\_1() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.15 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.27 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_rna() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.875 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.15 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.27 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function guanine\_forth\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_1();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function rna\_adenine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

default\_rna();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function rna\_cap\_1() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 7.734909192e+24;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function rna\_cap\_2() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 7.734909192e+24;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function rna\_guanine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

default\_rna();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz + molsizez;

}

function cytosine\_back\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_forth\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_3();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_back\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_forth\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_3();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_back\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_forth\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_3();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function default\_dna\_2() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.15 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.27 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.42 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_3() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.92 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.82 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.77 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function guanine\_back\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_2();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_forth\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_3();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_back\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_4();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function default\_dna\_4() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.92 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.82 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.77 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_5() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

cursor1y = (anchory - 0.47 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.43 \* molsizey) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.38 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.38 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.47 \* molsizey) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.72 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_6() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1y = (anchory - 0.47 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.43 \* molsizey) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.38 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.38 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.47 \* molsizey) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.72 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.72 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_7() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

cursor1y = (anchory - 0.47 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.43 \* molsizey) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.58 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.58 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.47 \* molsizey) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.32 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_8() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

tetracarbon();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1y = (anchory - 0.47 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.43 \* molsizey) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.58 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.58 \* molsizey) + 0;

cursor1z = (anchorz + 0.99 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.47 \* molsizey) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.32 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.32 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

}

}

function guanine\_back\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_4();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_back\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_4();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_back\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_4();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_up\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_6();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_up\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_5();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_up\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_6();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_up\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_5();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_down\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_8();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_down\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_7();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_down\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_8();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_down\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_7();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_up\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_6();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_up\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_5();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_up\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_6();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_up\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_5();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_down\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_8();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_down\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_7();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_down\_back() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_8();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_down\_forth() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_7();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_left\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_10();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function adenine\_left\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_9();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

heteronitrogen\_4();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.749 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.791 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['adenine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function default\_dna\_10() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.24 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.16 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.95 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.92 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.95 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.85768 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.95001 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.95213 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function default\_dna\_9() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.24 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.16 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.15 \* molsizey) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1y = (anchory - 0.1 \* molsizey) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.15 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.12 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.05768 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.152 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

}

function guanine\_left\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_10();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function guanine\_left\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_9();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.625 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

heteronitrogen\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.635 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['guanine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_left\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_10();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cytosine\_left\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.10389e+25;

molsizey = 2.10389e+25;

molsizez = 8.415581e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_9();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

heteronitrogen\_4();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.89 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['cytosine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_left\_down() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_10();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thymine\_left\_up() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 2.1038e+25;

molsizez = 8.41558e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

default\_dna\_9();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.555 \* molsizey) + 0;

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.545 \* molsizez) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.525 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.535 \* molsizez) + 0;

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.8759 \* molsizex) + 0;

cursor1z = (anchorz + 0.5857 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['thymine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function dna\_cap\_1() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_anion\_back();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function dna\_cap\_2() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 2.1038e+25;

molsizey = 8.41558e+25;

molsizez = 2.1038e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['molecule cap', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function carmine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.95 \* molsizez) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.975 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5365 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5365 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.5367 \* molsizey) + 0;

cursor1z = (anchorz + 0.935 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.535 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.0575 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

cursor1y = (anchory - 0.6375 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.0575 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

cursor1y = (anchory - 0.6375 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

cursor1y = (anchory - 0.6375 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.6375 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.135 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.125 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.175 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.325 \* molsizey) + 0;

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

hydrogen\_down();

}

list\_molecules[list\_molecules.length - 1] = ['carmine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function melanin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 8.044305559e+25;

molsizey = 3.093963676e+25;

molsizez = 8.044305559e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_6();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

carbon\_pyramid\_1();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.0575 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7135 \* molsizex) + 0;

cursor1z = (anchorz + 0.537 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.2735 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

oxygen\_forth\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.435 \* molsizex) + 0;

cursor1z = (anchorz + 0.2735 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

nitrophene\_1();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

my\_69\_hydrogen();

anchorx = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_up();

anchorx = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.557 \* molsizey) + 0;

hydrogen\_up();

anchorx = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.945 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['melanin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function wax() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 1.5469818e+26;

molsizez = 4.0859067e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.95 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.93572 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

oxygen\_horizontal();

cursor1y = (anchory - 0.93572 \* molsizey) + 0;

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

nitrophene\_1();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.93572 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.87 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

for (var count = 0; count < 13; count++) {

tetracarbon();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

hydrogen\_down();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.87 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

for (var count2 = 0; count2 < 13; count2++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.87 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

for (var count3 = 0; count3 < 13; count3++) {

my\_69\_hydrogen();

cursor1y = (cursor1y + 0.04 \* molsizey) + 0;

}

}

list\_molecules[list\_molecules.length - 1] = ['wax', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function TPP() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.96 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.97 \* molsizez) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

oxygen\_vertical();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.67351 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.94 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.92 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.595 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.595 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.465 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.505 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.57351 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.41325 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.41325 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.597 \* molsizex) + 0;

cursor1y = (anchory - 0.5375 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

heteronitrogen\_3();

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75123 \* molsizex) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.82517 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.24573 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['TPP', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function indigotine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

randomizer = math\_random\_int(1, 60);

if (randomizer <= 40) {

salt\_sodium();

} else {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_3();

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.74 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

allographene\_double\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.74 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75001 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.74 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25001 \* molsizex) + 0;

cursor1z = (anchorz + 0.74 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.59257 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

tetrasulphur();

cursor1z = (anchorz + 0.16573 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16573 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

tetrasulphur();

cursor1z = (anchorz + 0.89753 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1z = (anchorz + 0.86375 \* molsizez) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.89753 \* molsizez) + 0;

my\_69\_hydrogen();

}

}

list\_molecules[list\_molecules.length - 1] = ['indigotine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function thiamine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_1();

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

tetranitrogen\_cation();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.595 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.595 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.485 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

carbon\_pyramid\_2();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.465 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.505 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1x = (anchorx - 0.57351 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.41325 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.41325 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_63\_sulphur();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.597 \* molsizex) + 0;

cursor1y = (anchory - 0.5375 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

heteronitrogen\_3();

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.75123 \* molsizex) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.82517 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.24573 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['thiamine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cardiolipin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.0859067e+25;

molsizey = 4.640945515e+25;

molsizez = 4.764704062e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.482 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.464 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.446 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.428 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.392 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.374 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.518 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.536 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.554 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.572 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.608 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.626 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.356 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.338 \* molsizez) + 0;

for (var count = 0; count < 7; count++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count2 = 0; count2 < 2; count2++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count3 = 0; count3 < 4; count3++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.338 \* molsizez) + 0;

for (var count4 = 0; count4 < 7; count4++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count5 = 0; count5 < 2; count5++) {

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count6 = 0; count6 < 4; count6++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.54 \* molsizey) + 0;

cursor1z = (anchorz + 0.338 \* molsizez) + 0;

for (var count7 = 0; count7 < 7; count7++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count8 = 0; count8 < 1; count8++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count9 = 0; count9 < 7; count9++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.644 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.642 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.662 \* molsizez) + 0;

for (var count10 = 0; count10 < 7; count10++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count11 = 0; count11 < 1; count11++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count12 = 0; count12 < 7; count12++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.662 \* molsizez) + 0;

for (var count13 = 0; count13 < 7; count13++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count14 = 0; count14 < 1; count14++) {

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count15 = 0; count15 < 7; count15++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.54 \* molsizey) + 0;

cursor1z = (anchorz + 0.662 \* molsizez) + 0;

for (var count16 = 0; count16 < 7; count16++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count17 = 0; count17 < 1; count17++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count18 = 0; count18 < 7; count18++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.608 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.608 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.392 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.444 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.552 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.446 \* molsizez) + 0;

oxygen\_horizontal();

cursor1z = (anchorz + 0.554 \* molsizez) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.446 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.554 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.482 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.518 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.482 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.518 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.392 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.608 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.392 \* molsizez) + 0;

cursor1y = (anchory - 0.64 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.608 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.64 \* molsizey) + 0;

cursor1z = (anchorz + 0.395 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.604 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.385 \* molsizez) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

oxygen\_vertical();

cursor1z = (anchorz + 0.615 \* molsizez) + 0;

oxygen\_vertical();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.62 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.618 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.638 \* molsizez) + 0;

for (var count19 = 0; count19 < 7; count19++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count20 = 0; count20 < 1; count20++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count21 = 0; count21 < 7; count21++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.638 \* molsizez) + 0;

for (var count22 = 0; count22 < 7; count22++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count23 = 0; count23 < 1; count23++) {

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count24 = 0; count24 < 7; count24++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.86 \* molsizey) + 0;

cursor1z = (anchorz + 0.638 \* molsizez) + 0;

for (var count25 = 0; count25 < 7; count25++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count26 = 0; count26 < 1; count26++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

for (var count27 = 0; count27 < 7; count27++) {

hydrogen\_up();

cursor1z = (cursor1z + 0.015 \* molsizez) + 0;

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.368 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.352 \* molsizez) + 0;

for (var count28 = 0; count28 < 7; count28++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count29 = 0; count29 < 2; count29++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

allographene\_double\_6();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count30 = 0; count30 < 4; count30++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.9 \* molsizey) + 0;

cursor1z = (anchorz + 0.352 \* molsizez) + 0;

for (var count31 = 0; count31 < 7; count31++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count32 = 0; count32 < 2; count32++) {

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count33 = 0; count33 < 4; count33++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.86 \* molsizey) + 0;

cursor1z = (anchorz + 0.352 \* molsizez) + 0;

for (var count34 = 0; count34 < 7; count34++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count35 = 0; count35 < 1; count35++) {

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

cursor1x = (cursor1x + 0.1 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (cursor1x - 0.1 \* molsizex) + 0;

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

for (var count36 = 0; count36 < 7; count36++) {

hydrogen\_up();

cursor1z = (cursor1z - 0.015 \* molsizez) + 0;

}

}

list\_molecules[list\_molecules.length - 1] = ['cardiolipin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function pantothenic\_acid() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

nitrophene\_1();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.74 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.77 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.26 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.61 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.71 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.61 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.61 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.577 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['pantothenic acid', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function pyridoxine() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 5.25973825e+25;

molsizey = 3.712756412e+25;

molsizez = 7.425512824e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

heteronitrogen\_1();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_4();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allographene\_double\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

oxygen\_vertical();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['pyridoxine', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function tocopherol() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 1.856378206e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.11 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1z = (anchorz + 0.301 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.041 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.11 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

for (var count = 0; count < 14; count++) {

carbon\_pyramid\_3();

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

}

hydrogen\_forth();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

for (var count2 = 0; count2 < 14; count2++) {

cursor1z = (cursor1z + 0.04 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.39 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.71 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.575 \* molsizey) + 0;

cursor1z = (anchorz + 0.79 \* molsizez) + 0;

hydrogen\_up();

}

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.865 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.225 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.13 \* molsizex) + 0;

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_forth();

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

hydrogen\_down();

}

list\_molecules[list\_molecules.length - 1] = ['tocopherol', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function retinal() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 3.093963676e+25;

molsizey = 3.093963676e+25;

molsizez = 1.23758547e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.24 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.38 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.59 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.66 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

oxygen\_forth\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.065 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.88 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.335 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.287 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.735 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.687 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.187 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.785 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.737 \* molsizex) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.477 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.785 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.737 \* molsizex) + 0;

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.777 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1x = (anchorx - 0.775 \* molsizex) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.737 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.187 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['retinal', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function riboflavin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 9.28189103e+25;

molsizey = 3.093963676e+25;

molsizez = 9.28189103e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

heteronitrogen\_1();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

oxygen\_forth\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

heteronitrogen\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.93 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9453 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.935 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.07 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.835 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

oxygen\_horizontal();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

tetraphosphorus();

cursor1x = (anchorx - 0.1 \* molsizex) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_69\_hydrogen();

}

list\_molecules[list\_molecules.length - 1] = ['riboflavin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function ergocalciferol() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 9.28189103e+25;

molsizey = 3.093963676e+25;

molsizez = 1.546981838e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.42 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.28 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.58 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.43 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.63 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.665 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.415 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.42 \* molsizex) + 0;

cursor1y = (anchory - 0.415 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_down();

}

list\_molecules[list\_molecules.length - 1] = ['ergosterol', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function phylloquinone() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.331549147e+25;

molsizey = 3.712756412e+25;

molsizez = 2.042016026e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.12 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.89 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.71 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.415 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.39 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1z = (anchorz + 0.27 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.31 \* molsizez) + 0;

hydrogen\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['phylloquinone', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function folate() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 5.569134618e+25;

molsizey = 3.093963676e+25;

molsizez = 1.856378206e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

heteronitrogen\_1();

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.21 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

heteronitrogen\_1();

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.41 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.73 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.017 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.33 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.375 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.61 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.67 \* molsizez) + 0;

oxygen\_back\_2();

}

list\_molecules[list\_molecules.length - 1] = ['folate', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function moco() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 4.331549147e+25;

molsizey = 3.093963676e+25;

molsizez = 1.113826923e+26;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.05 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

heteronitrogen\_1();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.44 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.79 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.86 \* molsizez) + 0;

oxygen\_anion\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.16 \* molsizez) + 0;

nitrophene\_2();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

allographene\_double\_2();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_3();

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

nitrophene\_1();

cursor1z = (anchorz + 0.44 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.09 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.44 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

hydrogen\_down();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.22 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.37 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.44 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.51 \* molsizez) + 0;

sulphur\_anion();

cursor1z = (anchorz + 0.58 \* molsizez) + 0;

sulphur\_anion();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.857 \* molsizex) + 0;

cursor1y = (anchory - 0.85 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.55 \* molsizez) + 0;

molybdenum\_central();

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

oxygen\_anion\_forth();

}

list\_molecules[list\_molecules.length - 1] = ['moco', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function heme() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 5.878530986e+25;

molsizey = 4.640945515e+25;

molsizez = 5.878530986e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_6();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allonitrophene\_heme\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

nitrophene\_heme\_1();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

iron\_central();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

nitrophene\_heme\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allonitrophene\_heme\_1();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allonitrophene\_heme\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.375 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.375 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.13 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.12 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.13 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.12 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

cursor1y = (anchory - 0.45 \* molsizey) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.63 \* molsizex) + 0;

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1z = (anchorz + 0.93 \* molsizez) + 0;

hydrogen\_up();

}

list\_molecules[list\_molecules.length - 1] = ['heme', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cobalamin() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 7.425512824e+25;

molsizey = 7.425512824e+25;

molsizez = 7.425512824e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cobalamin\_part\_1();

cobalamin\_part\_2();

cobalamin\_part\_3();

list\_molecules[list\_molecules.length - 1] = ['cobalamin', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function heme\_c() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 5.878530986e+25;

molsizey = 4.640945515e+25;

molsizez = 5.878530986e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_6();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allonitrophene\_heme\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

nitrophene\_heme\_1();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

iron\_central();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

nitrophene\_heme\_2();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allonitrophene\_heme\_1();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

allonitrophene\_heme\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.375 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.615 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

hydrogen\_up();

cursor1y = (anchory - 0.375 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_down();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.68 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.13 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.28 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.82 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.18 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.13 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.12 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.1 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.15 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetracarbon();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

tetracarbon();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.78 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.13 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1y = (anchory - 0.7 \* molsizey) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.17 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.12 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_5();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_back\_2();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.9 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

my\_63\_sulphur();

cursor1z = (anchorz + 0.1513 \* molsizez) + 0;

my\_69\_hydrogen();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.85 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.9 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.83 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_forth();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.6775 \* molsizex) + 0;

cursor1z = (anchorz + 0.8715 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.7357 \* molsizex) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

my\_63\_sulphur();

cursor1z = (anchorz + 0.81 \* molsizez) + 0;

my\_69\_hydrogen();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.75 \* molsizey) + 0;

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.677 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.83 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.87 \* molsizez) + 0;

hydrogen\_forth();

}

}

list\_molecules[list\_molecules.length - 1] = ['heme c', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function cobalamin\_part\_1() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

carbon\_pyramid\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

allonitrophene\_heme\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

allonitrophene\_heme\_3();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

allographene\_double\_3();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

allographene\_double\_6();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cobalt\_central();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

allographene\_double\_6();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.6 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

nitrophene\_heme\_2();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

allonitrophene\_heme\_1();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

allographene\_double\_5();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

allographene\_double\_1();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

allographene\_double\_4();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

carbon\_pyramid\_3();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

carbon\_pyramid\_1();

}

}

function cobalamin\_part\_2() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

cursor1x = (anchorx - 0.65 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

hydrogen\_forth();

cursor1z = (anchorz + 0.77 \* molsizez) + 0;

cursor1x = (anchorx - 0.41 \* molsizex) + 0;

hydrogen\_forth();

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.55 \* molsizey) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_up();

}

if (unused\_variable == 5) {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.31 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.29 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.28 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.57 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.53 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.28 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.32 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.23 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.48 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.75 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.77 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.73 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.48 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.52 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.65 \* molsizey) + 0;

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.36 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.68 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.72 \* molsizez) + 0;

hydrogen\_forth();

}

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.38 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.19 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.36 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.64 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.64 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.74 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.71 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.67 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.71 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.76 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.81 \* molsizex) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.8 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.78 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.82 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.87 \* molsizex) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.92 \* molsizex) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.64 \* molsizex) + 0;

cursor1z = (anchorz + 0.79 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.62 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.64 \* molsizex) + 0;

cursor1z = (anchorz + 0.79 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.6 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.64 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.69 \* molsizex) + 0;

cursor1z = (anchorz + 0.8 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.74 \* molsizex) + 0;

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.7 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1z = (anchorz + 0.69 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1y = (anchory - 0.8 \* molsizey) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

oxygen\_forth\_2();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

hydrogen\_forth();

}

}

function cobalamin\_part\_3() {

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetracarbon();

cursor1x = (anchorx - 0.22 \* molsizex) + 0;

cursor1z = (anchorz + 0.29 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.18 \* molsizex) + 0;

my\_69\_hydrogen();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.25 \* molsizex) + 0;

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

oxygen\_back\_2();

cursor1x = (anchorx - 0.2 \* molsizex) + 0;

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

nitrophene\_1();

cursor1z = (anchorz + 0.11 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.15 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1y = (anchory - 0.15 \* molsizey) + 0;

hydrogen\_down();

cursor1y = (anchory - 0.2 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

my\_69\_hydrogen();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.2 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1x = (anchorx - 0.32 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.17 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.23 \* molsizez) + 0;

hydrogen\_forth();

}

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

cursor1y = (anchory - 0.25 \* molsizey) + 0;

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_78\_oxygen();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

tetraphosphorus();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.33 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.223 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.27 \* molsizex) + 0;

oxygen\_back\_2();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_1();

cursor1y = (anchory - 0.435 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.435 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.35 \* molsizez) + 0;

carbon\_pyramid\_4();

cursor1y = (anchory - 0.3 \* molsizey) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.437 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

my\_78\_oxygen();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.4 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.435 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.445 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.425 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.475 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

carbon\_pyramid\_3();

cursor1y = (anchory - 0.435 \* molsizey) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.435 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

my\_78\_oxygen();

cursor1z = (anchorz + 0.25 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

nitrophene\_2();

cursor1x = (anchorx - 0.49 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.49 \* molsizez) + 0;

allographene\_double\_1();

cursor1y = (anchory - 0.35 \* molsizey) + 0;

cursor1z = (anchorz + 0.47 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.495 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.53 \* molsizez) + 0;

allonitrophene\_heme\_4();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.45 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_2();

cursor1x = (anchorx - 0.495 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.57 \* molsizez) + 0;

allographene\_double\_3();

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.4 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

allographene\_double\_6();

cursor1z = (anchorz + 0.75 \* molsizez) + 0;

allographene\_double\_5();

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.65 \* molsizez) + 0;

hydrogen\_down();

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.55 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.6 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.525 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.825 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.875 \* molsizez) + 0;

hydrogen\_forth();

}

if (unused\_variable == 5) {

cursor1x = (anchorx - 0.4 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.85 \* molsizez) + 0;

carbon\_pyramid\_2();

cursor1y = (anchory - 0.45 \* molsizey) + 0;

cursor1x = (anchorx - 0.35 \* molsizex) + 0;

hydrogen\_up();

cursor1x = (anchorx - 0.375 \* molsizex) + 0;

cursor1y = (anchory - 0.3 \* molsizey) + 0;

cursor1z = (anchorz + 0.825 \* molsizez) + 0;

my\_69\_hydrogen();

cursor1z = (anchorz + 0.875 \* molsizez) + 0;

hydrogen\_forth();

}

}

}

function air\_argon() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

argon\_unbound\_();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_helium() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

randomizer = math\_random\_int(1, 1e+30);

if (randomizer <= (0.0002 / 100) \* 1e+30) {

lunar\_helium();

} else {

helium\_unbound\_();

}

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_krypton() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

krypton\_unbound\_();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_neon() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

neon\_unbound\_();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function air\_xenon() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

xenon\_unbound\_();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function bone() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

screen\_cells();

screen\_tissues();

if (make == true) {

make = false;

if (bvar == 1) {

vertebrate\_skeleton();

} else {

human\_skeleton();

}

if (make == true) {

make = false;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

thickabs = 1.361344017e+30;

thickabs2 = 0;

target2 = null;

target1 = ['artery', 'lymph', 'vein'];

layer();

if (make == false) {

make = false;

thickrel = null;

thickrel2 = null;

thickbod = null;

thickbod2 = null;

thickabs = 9.28189103e+28;

thickabs2 = 0;

target2 = null;

target1 = ['capillary', 'arteriole', 'venule'];

layer();

if (make == false) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 30) {

water\_molecule();

} else {

if (make == true) {

randomizer = math\_random\_int(1, 100);

if (randomizer <= 60) {

salt\_calcium();

} else {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

tetraphosphorus();

cursor1z = (anchorz + 0.3 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1z = (anchorz + 0.7 \* molsizez) + 0;

oxygen\_anion\_forth();

cursor1x = (anchorx - 0.7 \* molsizex) + 0;

cursor1z = (anchorz + 0.45 \* molsizez) + 0;

oxygen\_anion\_back();

cursor1x = (anchorx - 0.3 \* molsizex) + 0;

oxygen\_back\_2();

}

}

}

list\_molecules.push(['bone', anchorz, anchory, anchorx, anchorz + molsizez, anchory - molsizey, anchorx - molsizex]);

}

}

}

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function iron\_cofactor() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.9220687e+25;

molsizey = 1.9220687e+25;

molsizez = 1.9220687e+25;

loc1 = [anchorz, anchory, anchorx];

loc2 = [anchorz + molsizez, anchory - molsizey, anchorx - molsizex];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = (anchorx - 0.5 \* molsizex) + 0;

cursor1y = (anchory - 0.5 \* molsizey) + 0;

cursor1z = (anchorz + 0.5 \* molsizez) + 0;

iron\_central();

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function my\_69\_hydrogen() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

generate\_proton();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (218 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.59844 \* 8.190671898165e-29 - ((28.836 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function helium\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = ['proton', 'proton', 'neutron', 'neutron'];

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (24.58738 \* 8.190671898165e-29 - ((20.786 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 54.41776 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function hydrogen\_forth() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

generate\_proton();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (218 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.59844 \* 8.190671898165e-29 - ((28.836 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function hydrogen\_up() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

generate\_proton();

cursor1y = aty + 6.187927353e+22;

energy = (0 - (218 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.59844 \* 8.190671898165e-29 - ((28.836 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function hydrogen\_down() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

generate\_proton();

cursor1y = aty - 6.187927353e+22;

energy = (0 - (218 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.59844 \* 8.190671898165e-29 - ((28.836 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function hydrogen\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

generate\_proton();

cursor1z = atz + 6.187927353e+22;

energy = 0 - (13.59844 \* 8.190671898165e-29 - ((28.836 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function lunar\_helium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = ['proton', 'proton', 'neutron'];

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (24.58738 \* 8.190671898165e-29 - ((20.786 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 54.41776 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function lists\_get\_random\_item(list, remove) {

var x = Math.floor(Math.random() \* list.length);

if (remove) {

return list.splice(x, 1)[0];

} else {

return list[x];

}

}

function nucleus() {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

hadroncount = hadronlist.length;

while (hadronlist.length != 0) {

var repeat\_end2 = Math.round(Math.pow(hadroncount, 1 / 3));

for (var count2 = 0; count2 < repeat\_end2; count2++) {

var repeat\_end = Math.round(Math.pow(hadroncount, 1 / 3));

for (var count = 0; count < repeat\_end; count++) {

if (hadronlist.length != 0) {

hadron = lists\_get\_random\_item(hadronlist, true);

if (hadron == 'proton') {

generate\_proton();

} else {

Generate\_neutron();

}

}

cursor1x = cursor1x - 52597382506729650000;

}

cursor1x = atx;

cursor1y = cursor1y - 52597382506729650000;

}

cursor1y = aty;

cursor1z = cursor1z + 52597382506729650000;

}

}

function my\_32\_beryllium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 4; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 5; count2++) {

hadronlist.push('neutron');

}

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (29.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (9.32269 \* 8.190671898165e-29 - ((16.443 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (29.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.21115 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 1.187927353e+22;

energy = 0 - 153.8962 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 217.71858 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_46\_lithium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 3; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 4; count4++) {

hadronlist.push('neutron');

}

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (52.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.39171 \* 8.190671898165e-29 - ((24.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 3e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75.64009 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 122.45435 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function lithium\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 3; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 4; count6++) {

hadronlist.push('neutron');

}

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (5.39171 \* 8.190671898165e-29 - ((24.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75.64009 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 122.45435 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_lithium() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922e+25;

molsizey = 1.922e+25;

molsizez = 1.922e+25;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + molsizez, cursor1y - molsizey, cursor1x - molsizex];

exempt = [null];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = anchorx - molsizex \* 0.5;

cursor1y = anchory - molsizey \* 0.5;

cursor1z = anchorz + molsizez \* 0.5;

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 3; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 4; count10++) {

hadronlist.push('neutron');

}

nucleus();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (75.64009 \* 8.190671898165e-29 - ((24.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 122.45435 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function my\_42\_boron() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 5; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 6; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (145 / 6.0221409e+23) \* 5.11221307704105e-7) - (8.29803 \* 8.190671898165e-29 - ((11.087 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (145 / 6.0221409e+23) \* 5.11221307704105e-7) - 25.15484 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 1.187927353e+22;

energy = 0 - 37.93064 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 259.37521 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 340.2258 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function borophene() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 5; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 6; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (145 / 6.0221409e+23) \* 5.11221307704105e-7) - (8.29803 \* 8.190671898165e-29 - ((11.087 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (145 / 6.0221409e+23) \* 5.11221307704105e-7) - 25.15484 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (145 / 6.0221409e+23) \* 5.11221307704105e-7) - 37.93064 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 259.37521 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 340.2258 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_62\_carbon() {

tetracarbon();

}

function carbon\_unbound\_() {

anchorx = cursor1x;

anchory = cursor1y;

anchorz = cursor1z;

molsizex = 1.922e+25;

molsizey = 1.922e+25;

molsizez = 1.922e+25;

loc1 = [cursor1z, cursor1y, cursor1x];

loc2 = [cursor1z + molsizez, cursor1y - molsizey, cursor1x - molsizex];

exempt = [null];

make = true;

screen\_molecules();

if (make == true) {

cursor1x = anchorx - molsizex \* 0.5;

cursor1y = anchory - molsizey \* 0.5;

cursor1z = anchorz + molsizez \* 0.5;

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 6; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 6; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 64.4939 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

cursor1x = anchorx;

cursor1y = anchory;

cursor1z = anchorz;

}

function diamond\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 6; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 6; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx - 6.187927353e+22;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty - 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function diamond\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 6; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 6; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx - 6.187927353e+22;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty - 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function diamond\_3() {

diamond\_1();

}

function diamond\_4() {

diamond\_2();

}

function graphene\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 6; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 6; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((8.517 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 64.4939 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function graphene\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 6; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 6; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((8.517 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 64.4939 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function graphene\_anion() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 6; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 6; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((8.517 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 64.4939 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1.2621226 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function tetracarbon() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 6; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 6; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1y = aty - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_62\_carbon\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 6; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 6; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 6; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 6; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz + 6.187927353e+22;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 6; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 6; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 6; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 6; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz + 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_4() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 6; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 6; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_5() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 6; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 6; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allographene\_double\_6() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 6; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 6; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x - 6187927353732900000;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 6.187927353e+22;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_double\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 6; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 6; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1y = aty + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty;

cursor1x = cursor1x - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_double\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 6; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 6; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1y = aty - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (171.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty;

cursor1x = cursor1x - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_pyramid\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count19 = 0; count19 < 6; count19++) {

hadronlist.push('proton');

}

for (var count20 = 0; count20 < 6; count20++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_pyramid\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count21 = 0; count21 < 6; count21++) {

hadronlist.push('proton');

}

for (var count22 = 0; count22 < 6; count22++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1y = aty;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_pyramid\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count23 = 0; count23 < 6; count23++) {

hadronlist.push('proton');

}

for (var count24 = 0; count24 < 6; count24++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1y = aty;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function carbon\_pyramid\_4() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count25 = 0; count25 < 6; count25++) {

hadronlist.push('proton');

}

for (var count26 = 0; count26 < 6; count26++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.2603 \* 8.190671898165e-29 - ((6.115 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 24.38332 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.8878 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (188.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 64.4939 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 392.087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.99334 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_49\_nitrogen() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 7; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 7; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allonitrophene\_cation() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 7; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 7; count4++) {

hadronlist.push('neutron');

}

cursor1z = atz + 6.187927353e+22;

cursor1y = aty;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (29.6013 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 97.8902 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function heme\_nitrophene\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 7; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 7; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function heteronitrogen\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 7; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 7; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function heteronitrogen\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 7; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 7; count10++) {

hadronlist.push('neutron');

}

cursor1z = atz;

cursor1y = aty;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function heteronitrogen\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 7; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 7; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function heteronitrogen\_4() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 7; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 7; count14++) {

hadronlist.push('neutron');

}

cursor1z = atz;

cursor1y = aty;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrogen\_back\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 7; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 7; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.777982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (244.604831858 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrogen\_forth\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 7; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 7; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.777982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (244.612831858 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrogen\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count19 = 0; count19 < 7; count19++) {

hadronlist.push('proton');

}

for (var count20 = 0; count20 < 7; count20++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 29.6013 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz - 1e+22;

energy = 0 - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrophene\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count21 = 0; count21 < 7; count21++) {

hadronlist.push('proton');

}

for (var count22 = 0; count22 < 7; count22++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1x = atx + 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrophene\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count23 = 0; count23 < 7; count23++) {

hadronlist.push('proton');

}

for (var count24 = 0; count24 < 7; count24++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function tetranitrogen\_cation() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count25 = 0; count25 < 7; count25++) {

hadronlist.push('proton');

}

for (var count26 = 0; count26 < 7; count26++) {

hadronlist.push('neutron');

}

cursor1z = atz + 6.187927353e+22;

cursor1y = aty;

cursor1x = atx;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (29.6013 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 97.8902 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allonitrophene\_heme\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 7; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 7; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx - 6.187927353e+22;

cursor1y = aty;

cursor1z = atz;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allonitrophene\_heme\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 7; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 7; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx - 6.187927353e+22;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allonitrophene\_heme\_3() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 7; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 7; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function allonitrophene\_heme\_4() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 7; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 7; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx - 6.187927353e+22;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (147.603982301 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.4735 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrophene\_heme\_1() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 7; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 7; count10++) {

hadronlist.push('neutron');

}

cursor1y = aty;

cursor1x = atx + 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nitrophene\_heme\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 7; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 7; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1x = atx - 6.187927353e+22;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - (14.53414 \* 8.190671898165e-29 - ((29.124 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (139.282318584 / 6.0221409e+23) \* 5.11221307704105e-7) - 29.6013 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (80.2831858407 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.44924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.4735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.8902 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 552.0718 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 667.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_78\_oxygen() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 8; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 8; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - (((0 - (5466.2778 - 6008.996644)) / 6.0221409e+23) \* 5.11221307704105e-10 + ((17.70393152 / 6.0221409e+23) \* 5.11221307704105e-10) \* (material\_temperature\_\_kelvins - 273.15)));

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_anion\_back() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 8; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 8; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1.4611136 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_anion\_forth() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 8; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 8; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1.4611136 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_back\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 8; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 8; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (3.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_cobalamin() {

oxygen\_vertical();

}

function oxygen\_forth\_2() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 8; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 8; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (3.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_horizontal() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 8; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 8; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

cursor1z = atz;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - (((0 - (5466.2778 - 6008.996644)) / 6.0221409e+23) \* 5.11221307704105e-10 + ((17.70393152 / 6.0221409e+23) \* 5.11221307704105e-10) \* (material\_temperature\_\_kelvins - 273.15)));

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_ozone\_middle() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 8; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 8; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (35.1173 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (3.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 77.41353 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_vertical() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 8; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 8; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty + 6.187927353e+22;

cursor1z = atz;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (13.61806 \* 8.190671898165e-29 - (((0 - (5466.2778 - 6008.996644)) / 6.0221409e+23) \* 5.11221307704105e-10 + ((17.70393152 / 6.0221409e+23) \* 5.11221307704105e-10) \* (material\_temperature\_\_kelvins - 273.15)));

add\_electron();

cursor1y = aty - 6.187927353e+22;

energy = (0 - (245.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function oxygen\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 8; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 8; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (13.61806 \* 8.190671898165e-29 - ((29.378 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35.1173 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77.41353 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 113.899 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.1197 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 739.29 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 871.4101 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_66\_fluorine() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 9; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 10; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (78.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (17.42282 \* 8.190671898165e-29 - ((31.304 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34.97082 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 62.7084 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 87.1398 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 114.2428 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 157.1651 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 185.186 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 953.9112 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1103.1176 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_72\_magnesium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 12; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 12; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (148.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.64624 \* 8.190671898165e-29 - ((24.869 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (148.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 15.03528 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 80.1437 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 109.2655 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 141.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 186.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 225.02 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 367.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.805 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1962.665 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_73\_sodium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 11; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 12; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (38.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.13908 \* 8.190671898165e-29 - ((28.23 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47.2864 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 71.62 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 98.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 172.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 208.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 264.25 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 299.864 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1465.121 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1648.702 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_76\_aluminium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 13; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 14; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (266.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.98577 \* 8.190671898165e-29 - ((24.2 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (266.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.82856 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1y = aty + 6.187927353e+22;

energy = (0 - (266.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 28.44765 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 119.992 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 153.825 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 190.49 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 241.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 284.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 398.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2085.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2304.141 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function aluminium\_metallic\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 13; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 14; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (98 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.98577 \* 8.190671898165e-29 - ((24.2 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (98 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.82856 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1y = aty + 6.187927353e+22;

energy = (0 - (98 / 6.0221409e+23) \* 5.11221307704105e-7) - 28.44765 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 119.992 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 153.825 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 190.49 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 241.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 284.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 398.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2085.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2304.141 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function magnesium\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 12; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 12; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (7.64624 \* 8.190671898165e-29 - ((24.869 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15.03528 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 80.1437 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 109.2655 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 141.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 186.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 225.02 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 367.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.805 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1962.665 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function neon\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 10; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 10; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (21.5646 \* 8.190671898165e-29 - ((20.786 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 40.96328 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 63.45 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 126.21 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 157.93 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 207.2759 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 239.0989 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1195.8286 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1362.1995 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_magnesium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 12; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 12; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80.1437 \* 8.190671898165e-29 - ((24.869 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 109.2655 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 141.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 186.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 225.02 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 367.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.805 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1962.665 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_sodium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 11; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 12; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (47.2864 \* 8.190671898165e-29 - ((28.23 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 71.62 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 98.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 172.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 208.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 264.25 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 299.864 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1465.121 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1648.702 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_63\_sulphur() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 16; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 16; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (214.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (10.36001 \* 8.190671898165e-29 - ((22.75 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (214.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 23.3379 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34.79 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47.222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72.5945 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 88.053 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 280.948 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379.55 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 447.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 504.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 652.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 707.01 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3223.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3494.1892 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_68\_phosphorus() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 15; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 16; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - (10.48669 \* 8.190671898165e-29 - ((21.19 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 19.7694 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1y = aty - 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 30.2027 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.4439 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 65.0251 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 220.421 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 263.57 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 309.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 372.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 424.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 479.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 560.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 611.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2816.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3069.842 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_77\_silicon() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 14; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 14; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (552.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (8.15169 \* 8.190671898165e-29 - ((19.789 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (552.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.34585 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty - 6.187927353e+22;

energy = (0 - (552.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 33.49302 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty + 6.187927353e+22;

energy = (0 - (552.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 45.14181 \* 8.190671898165e-29;

add\_electron();

cursor1y = aty;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 166.767 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 205.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 246.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 303.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 351.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 401.37 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 476.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 523.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2437.63 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2673.182 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function phosphorus\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 15; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 16; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (10.48669 \* 8.190671898165e-29 - ((21.19 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19.7694 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.2027 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.4439 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 65.0251 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 220.421 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 263.57 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 309.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 372.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 424.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 479.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 560.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 611.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2816.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3069.842 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function silicon\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 14; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 14; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (8.15169 \* 8.190671898165e-29 - ((19.789 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16.34585 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 33.49302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45.14181 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 166.767 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 205.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 246.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 303.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 351.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 401.37 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 476.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 523.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2437.63 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2673.182 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function sulphur\_anion() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 16; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 16; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

cursor1z = atz;

energy = (0 - (214.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (10.36001 \* 8.190671898165e-29 - ((22.75 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 23.3379 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34.79 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47.222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72.5945 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 88.053 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 280.948 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379.55 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 447.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 504.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 652.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 707.01 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3223.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3494.1892 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2.077 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function sulphur\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 16; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 16; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (10.36001 \* 8.190671898165e-29 - ((22.75 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23.3379 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34.79 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47.222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72.5945 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 88.053 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 280.948 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379.55 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 447.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 504.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 652.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 707.01 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3223.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3494.1892 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function tetraphosphorus() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 15; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 16; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - (10.48669 \* 8.190671898165e-29 - ((21.19 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 19.7694 \* 8.190671898165e-29;

add\_electron();

cursor1z = atz;

cursor1x = atx - 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 30.2027 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx + 6.187927353e+22;

energy = (0 - (351.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 51.4439 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (111.875 / 6.0221409e+23) \* 5.11221307704105e-7) - 65.0251 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 220.421 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 263.57 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 309.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 372.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 424.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 479.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 560.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 611.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2816.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3069.842 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function tetrasulphur() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 16; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 16; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (276.35 / 6.0221409e+23) \* 5.11221307704105e-7) - (10.36001 \* 8.190671898165e-29 - ((22.75 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (276.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 23.3379 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (276.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 34.79 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (14.85 / 6.0221409e+23) \* 5.11221307704105e-7) - 47.222 \* 8.190671898165e-29;

add\_electron();

cursor1x = atx;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (276.35 / 6.0221409e+23) \* 5.11221307704105e-7) - 72.5945 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (14.85 / 6.0221409e+23) \* 5.11221307704105e-7) - 88.053 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 280.948 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 328.75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379.55 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 447.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 504.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 652.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 707.01 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3223.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3494.1892 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_60\_chlorine() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 17; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 18; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (26.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (12.96764 \* 8.190671898165e-29 - ((33.949 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23.814 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 39.61 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 53.4652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 67.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.03 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 114.1958 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 348.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 400.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 455.63 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 529.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 591.99 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 656.71 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 749.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 809.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3658.521 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3946.296 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_71\_potassium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 19; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 20; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (28.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (4.34066 \* 8.190671898165e-29 - ((29.6 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31.63 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45.806 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 60.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 82.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 117.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 154.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 175.8174 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 503.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 629.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 714.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 786.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 861.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 968 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1033.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4610.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4934.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function argon\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 18; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 22; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (15.75962 \* 8.190671898165e-29 - ((20.786 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 27.62967 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 40.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 59.81 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75.02 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 91.009 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.323 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 143.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 422.45 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 478.69 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 538.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 618.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 686.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 755.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 854.77 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 918.03 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4120.8857 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4426.2296 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_chloride() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 17; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 18; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (12.96764 \* 8.190671898165e-29 - ((33.949 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23.814 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 39.61 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 53.4652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 67.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97.03 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 114.1958 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 348.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 400.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 455.63 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 529.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 591.99 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 656.71 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 749.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 809.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3658.521 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3946.296 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3.612725 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_potassium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 19; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 20; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (31.63 \* 8.190671898165e-29 - ((29.6 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45.806 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 60.91 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 82.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 117.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 154.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 175.8174 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 503.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 564.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 629.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 714.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 786.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 861.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 968 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1033.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4610.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4934.046 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_47\_scandium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 21; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 24; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (428.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.5615 \* 8.190671898165e-29 - ((25.52 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (428.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 12.79967 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24.75666 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 73.4894 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 91.65 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 110.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 138 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 158.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 180.03 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 225.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 249.798 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 687.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 756.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 830.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 927.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1009 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1094 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1213 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1287.97 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5674.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6033.712 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_59\_vanadium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 23; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 28; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (398.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.7462 \* 8.190671898165e-29 - ((24.89 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (398.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 14.66 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 29.311 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 46.709 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 65.2817 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 128.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 150.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 173.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 205.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 230.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 255.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 308.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 336.277 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 896 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 976 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1060 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1168 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1260 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1355 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1486 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1569.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6851.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7246.12 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_70\_titanium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 22; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 26; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (416.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.8281 \* 8.190671898165e-29 - ((25.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (416.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 13.5755 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 27.4917 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 43.2672 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 119.53 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 140.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 170.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 192.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 215.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.07 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 291.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 787.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 863.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 941.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1044 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1131 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1221 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1346 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1425.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6249 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6625.82 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_74\_calcium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 20; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 20; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (218.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.11316 \* 8.190671898165e-29 - ((25.929 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (218.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 11.87172 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 50.9131 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 67.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 84.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 127.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 147.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 188.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 211.275 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 591.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 657.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 726.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 817.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 894.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 974 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1157.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5128.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5469.864 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_calcium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 20; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 20; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (50.9131 \* 8.190671898165e-29 - ((25.929 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 67.27 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 84.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 127.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 147.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 188.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 211.275 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 591.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 657.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 726.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 817.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 894.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 974 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1087 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1157.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5128.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5469.864 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_58\_chromium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 24; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 28; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (181.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.7665 \* 8.190671898165e-29 - ((23.35 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (181.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.4857 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 49.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 69.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 90.6349 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 160.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 184.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 209.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 244.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 270.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 298 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 354.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 384.168 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1010.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1097 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1299 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1396 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1496 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1634 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1721.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7481.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7894.81 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_67\_manganese() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 25; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 30; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (156.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.43402 \* 8.190671898165e-29 - ((26.32 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (156.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 15.63999 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 33.668 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 95.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 119.203 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 194.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 221.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 248.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 286 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 314.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 343.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 403 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 435.163 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1134.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1224 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1317 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1437 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1539 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1644 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1788 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1879.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8140.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8571.94 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_75\_iron() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 26; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 30; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (163.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.9024 \* 8.190671898165e-29 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (163.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.1878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 151.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 233.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 262.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 290.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 361 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 457 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.256 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1266 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1358 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1456 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1582 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1689 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1799 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1950 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2023 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8828 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9277.69 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function iron\_central() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 26; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 30; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (30.652 \* 8.190671898165e-29 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 151.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 233.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 262.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 290.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 361 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 457 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.256 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1266 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1358 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1456 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1582 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1689 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1799 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1950 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2023 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8828 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9277.69 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function iron\_horizontal() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 26; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 30; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx + 6.187927353e+22;

cursor1y = aty;

cursor1z = atz;

energy = (0 - (124.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.9024 \* 8.190671898165e-29 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (124.5 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.1878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 151.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 233.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 262.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 290.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 361 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 457 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.256 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1266 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1358 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1456 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1582 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1689 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1799 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1950 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2023 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8828 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9277.69 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function iron\_metallic\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 26; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 30; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (174.8 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.9024 \* 8.190671898165e-29 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (174.8 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.1878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 151.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 233.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 262.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 290.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 361 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 457 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.256 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1266 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1358 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1456 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1582 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1689 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1799 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1950 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2023 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8828 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9277.69 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function iron\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 26; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 30; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (7.9024 \* 8.190671898165e-29 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 16.1878 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.652 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 124.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 151.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 233.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 262.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 290.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 330.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 361 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 392.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 457 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 489.256 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1266 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1358 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1456 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1582 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1689 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1799 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1950 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2023 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8828 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9277.69 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_chromium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count15 = 0; count15 < 24; count15++) {

hadronlist.push('proton');

}

for (var count16 = 0; count16 < 28; count16++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (30.96 \* 8.190671898165e-29 - ((23.35 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 49.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 69.46 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 90.6349 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 160.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 184.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 209.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 244.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 270.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 298 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 354.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 384.168 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1010.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1097 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1299 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1396 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1496 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1634 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1721.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7481.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7894.81 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_manganese() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count17 = 0; count17 < 25; count17++) {

hadronlist.push('proton');

}

for (var count18 = 0; count18 < 30; count18++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (33.668 \* 8.190671898165e-29 - ((26.32 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 95.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 119.203 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 194.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 221.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 248.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 286 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 314.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 343.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 403 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 435.163 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1134.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1224 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1317 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1437 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1539 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1644 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1788 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1879.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8140.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8571.94 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_48\_cobalt() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 27; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 32; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (122.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.881 \* 8.190671898165e-29 - ((24.81 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (122.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 17.083 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 33.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 102 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 128.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 157.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 186.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 275.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 305 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 336 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 411 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 444 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 511.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 546.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1397.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1504.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1603 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1846 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1962 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2119 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2219 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9544.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10012.12 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_54\_copper() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 29; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 34; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (97.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.72638 \* 8.190671898165e-29 - ((24.44 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (97.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 20.2924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 36.841 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 57.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 103 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 139 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 166 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 199 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 232 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 369 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 401 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 435 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 484 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 520 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 557 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 633 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 670.588 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1697 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1804 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1916 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2060 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2182 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2308 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2478 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2587.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11062.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11567.617 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_56\_nickel() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 28; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 30; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (146.25 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.6398 \* 8.190671898165e-29 - ((26.07 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (146.25 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.16884 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35.19 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 76.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 133 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 162 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 193 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 224.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 321 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 352 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 384 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 430 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 464 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 499 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 571.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 607.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1541 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1648 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1756 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1894 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2011 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2131 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2295 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2399.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10288.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10775.4 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function cobalt\_central() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 27; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 32; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty - 6.187927353e+22;

cursor1z = atz;

energy = (0 - (122.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (17.083 \* 8.190671898165e-29 - ((24.81 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 1e+22;

energy = 0 - 33.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 102 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 128.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 157.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 186.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 275.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 305 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 336 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 379 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 411 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 444 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 511.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 546.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1397.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1504.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1603 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1735 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1846 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1962 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2119 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2219 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9544.1 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10012.12 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function copper\_metallic\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 29; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 34; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (150.15 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.72638 \* 8.190671898165e-29 - ((24.44 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (150.15 / 6.0221409e+23) \* 5.11221307704105e-7) - 20.2924 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 36.841 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 57.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 103 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 139 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 166 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 199 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 232 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 369 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 401 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 435 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 484 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 520 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 557 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 633 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 670.588 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1697 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1804 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1916 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2060 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2182 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2308 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2478 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2587.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11062.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11567.617 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function nickel\_metallic\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count11 = 0; count11 < 28; count11++) {

hadronlist.push('proton');

}

for (var count12 = 0; count12 < 30; count12++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (185.2 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.6398 \* 8.190671898165e-29 - ((26.07 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (185.2 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.16884 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35.19 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 54.9 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 76.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 133 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 162 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 193 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 224.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 321 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 352 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 384 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 430 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 464 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 499 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 571.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 607.06 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1541 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1648 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1756 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1894 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2011 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2131 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2295 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2399.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10288.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10775.4 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_copper() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count13 = 0; count13 < 29; count13++) {

hadronlist.push('proton');

}

for (var count14 = 0; count14 < 34; count14++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (36.841 \* 8.190671898165e-29 - ((24.44 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 57.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 103 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 139 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 166 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 199 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 232 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 265.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 369 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 401 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 435 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 484 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 520 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 557 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 633 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 670.588 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1697 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1804 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1916 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2060 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2182 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2308 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2478 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2587.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11062.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11567.617 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_25\_germanium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 32; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 42; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (416.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.8994 \* 8.190671898165e-29 - ((23.222 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (416.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 15.93462 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34.2241 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45.7131 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 93.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24320 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31450 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35410 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39270 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50930 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54740 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (63950 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (68190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (71790 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5043.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5163.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5283.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5360.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5470.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5700.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5850.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6075.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6275.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14340 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14940 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_44\_gallium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 31; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 38; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (39.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.9993 \* 8.190671898165e-29 - ((25.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (39.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 20.5142 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30.71 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 87.045789767 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 114.012258678 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16390 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20360 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23540 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30780 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34350 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49090 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57870 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (61750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (65320 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73880 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (77890 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4843.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4963.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5083.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5160.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5270.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5500.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5650.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5875.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13532 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14082 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_55\_zinc() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 30; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 34; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (38.75 / 6.0221409e+23) \* 5.11221307704105e-7) - (9.3942 \* 8.190671898165e-29 - ((25.39 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (38.75 / 6.0221409e+23) \* 5.11221307704105e-7) - 17.9644 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 39.723 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 59.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 82.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 134 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 174 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 203 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 238 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 274 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 310.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 419.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 454 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 490 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 542 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 579 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 619 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 698 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 738 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1856 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4643.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4763.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4883.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5010.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5150.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5300.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5450.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12762.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13252 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_zinc() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 30; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 34; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (39.723 \* 8.190671898165e-29 - ((25.39 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 59.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 82.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 108 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 134 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 174 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 203 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 238 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 274 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 310.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 419.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 454 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 490 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 542 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 579 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 619 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 698 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 738 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1856 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4643.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4763.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4883.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5010.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5150.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5300.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5450.91341018 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12762.38 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13252 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_11\_selenium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 34; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 46; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (177.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (9.75238 \* 8.190671898165e-29 - ((25.363 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (177.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 21.19 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3058.3 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4143.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6589.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7895 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14990 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28080 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33080 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36950 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49880 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62760 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (66900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (71380 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (77000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (81610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2193.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2193.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2193.5 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4387.53674419 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15996.408 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16736 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_27\_arsenic() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 33; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 42; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (235.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (9.7886 \* 8.190671898165e-29 - ((24.64 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (235.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.633 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 28.351 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 50.13 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 62.63 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11690 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17370 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23830 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28560 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40330 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44380 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56690 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (60670 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (64920 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (70330 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (74680 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (78540 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1785.37249336 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1785.37249336 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3985.41209302 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15127.272 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15824 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_30\_bromine() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 35; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 44; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (96.935 / 6.0221409e+23) \* 5.11221307704105e-7) - (11.81381 \* 8.190671898165e-29 - ((36.057 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 21.8 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3364.5 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4610.3 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8431.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9941 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18580 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29040 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37920 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42070 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46410 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55670 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (69110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73430 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (78130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (83940 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1945.95734614 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1945.95734614 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1945.95734614 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1945.95734614 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4595.86697674 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16892.228 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17676 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_57\_rubidium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 37; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 48; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (9.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (4.17713 \* 8.190671898165e-29 - ((31.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 27.285 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3786.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5037 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6603 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7999 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9520 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12810 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14530 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26740 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30210 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34350 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (63100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (68100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (82700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (87300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2315.90675838 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5230.25767442 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19760.108 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20636 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function krypton\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count9 = 0; count9 < 36; count9++) {

hadronlist.push('proton');

}

for (var count10 = 0; count10 < 48; count10++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - (13.99961 \* 8.190671898165e-29 - ((20.786 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - 24.35985 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3457.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4906 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6242 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7573 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10530 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22480 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29720 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37730 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43030 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47470 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (61750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (75740 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80240 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (85180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2029.71812789 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2029.71812789 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2029.71812789 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2029.71812789 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2029.71812789 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4709.95953488 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17812.826 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19642 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_50\_yttrium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 39; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 50; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (469.75 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.2171 \* 8.190671898165e-29 - ((26.53 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (469.75 / 6.0221409e+23) \* 5.11221307704105e-7) - 12.24 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1980.3 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5847.7 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7270 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8818 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10620 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14050 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17920 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36090 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (60200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (65300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (70700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (76200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (81700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2484.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4969.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6625.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6625.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19876.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21705 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_61\_zirconium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 40; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 50; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (514.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.6339 \* 8.190671898165e-29 - ((25.36 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (514.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 13.13 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2236 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3320.87 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7752 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9299 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10810 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14760 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20730 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22790 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (60000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (66600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (71900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (77500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (83300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1366.86 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2733.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5467.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21869.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22781 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_64\_strontium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 38; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 50; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (208.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (5.6949 \* 8.190671898165e-29 - ((26.4 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (208.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 11.03013 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4137.63 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5430.2 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6822 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8491 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10030 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11690 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15280 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31270 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34930 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (59000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (64200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (69700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (74700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (89900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2358.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4717.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19870.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20657 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_23\_molybdenum() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 42; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 56; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.09243 \* 8.190671898165e-29 - ((24.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.16 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2618 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3891 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6641 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15840 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (67700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (85900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 25932.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 27013 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_45\_niobium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 41; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 52; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (507.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (6.75885 \* 8.190671898165e-29 - ((24.6 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (507.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 14.32 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2416 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3629 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4880 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9848 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11490 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15360 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17370 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19320 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (61400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (66400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (78700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (84600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1492.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1492.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2985.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5971.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24883 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function molybdenum\_central() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 42; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 56; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.09243 \* 8.190671898165e-29 - ((24.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (35.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (2618 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (3891 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6641 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15840 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (67700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (85900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 25932.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 27013 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function tetramolybdenum() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 42; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 56; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz - 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.09243 \* 8.190671898165e-29 - ((24.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz + 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (35.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (2618 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1z = atz;

cursor1x = atx + 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (3891 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = atx - 6.187927353e+22;

energy = (0 - (361.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (5250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = (0 - (35.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (6641 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

energy = 0 - (12120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15840 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (67700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (73100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (80000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (85900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1620.78 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3241.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6483.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 25932.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 27013 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_10\_palladium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 46; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 60; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (117 / 6.0221409e+23) \* 5.11221307704105e-7) - (8.3369 \* 8.190671898165e-29 - ((25.98 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (117 / 6.0221409e+23) \* 5.11221307704105e-7) - 19.43 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3177 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4438 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5886 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8114 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9745 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11580 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15430 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25090 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (78200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (83800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (90000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1908.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3816.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31805 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_12\_silver() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 47; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 60; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (106.5 / 6.0221409e+23) \* 5.11221307704105e-7) - ((731 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.35 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (106.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (2072.93 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3358 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4728 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6272 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7912 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10230 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12060 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (85400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (91300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1984.26 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3968.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7937.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31748.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 33071 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_2\_rhodium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 45; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 58; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (131.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.4589 \* 8.190671898165e-29 - ((24.98 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (131.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 18.08 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2997 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4052 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6079 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7719 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9359 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13030 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24320 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (71300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (76600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (82700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (88900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1773.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3547.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7095.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 28383.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 29566 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_4\_ruthenium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 44; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 58; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (235.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (7.3605 \* 8.190671898165e-29 - ((24.06 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (235.65 / 6.0221409e+23) \* 5.11221307704105e-7) - 16.76 \* 8.190671898165e-29;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2747 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4342 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5693 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7333 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8973 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17210 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (64600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (69800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (75600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (81500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (87300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 1761.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3522.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7044.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 28179.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 29354 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_14\_cadmium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 48; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 66; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (71 / 6.0221409e+23) \* 5.11221307704105e-7) - ((867.77 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.02 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (71 / 6.0221409e+23) \* 5.11221307704105e-7) - (1631.4 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3615.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4921 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6551 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8394 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14470 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16690 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18810 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21030 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (92900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2121.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4243.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8487.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 33949.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35364 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_15\_antimony() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 51; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 70; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (126.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((830.58 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.23 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (126.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1604.2 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2443.35 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4226.4 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5307 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9601 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11290 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13410 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15630 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17850 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20650 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22960 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2364.42 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4728.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9457.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 37830.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 39407 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_16\_indium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 49; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 64; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (114.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((558.3 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.74 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (114.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1820.72 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2705.85 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5350 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6686 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8684 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10520 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15050 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17170 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19390 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21810 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2141.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4282.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8564.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 34256.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35684 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_29\_tin() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 50; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 70; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (302.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((708.58 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.77 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (302.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1411.88 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2943.4 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3931 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7432 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9070 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10890 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13030 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15050 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20070 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22380 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2221.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4443.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8887.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 35550.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 37032 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_17\_iodine() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 53; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 74; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (92 / 6.0221409e+23) \* 5.11221307704105e-7) - ((1008.39 / 6.0221409e+23) \* 5.11221307704105e-7 - ((36.888 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - (1845.89 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2853.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3893.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4971 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7179 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8453 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19010 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21310 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23830 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 40548.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 42238 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_5\_tellurium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 52; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 74; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (145.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((869.3 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.73 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (145.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1795 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2686 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3610.05 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5722 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6667 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18440 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20740 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2448.54 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4897.08 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9794.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 39176.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 40809 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function salt\_iodine() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 53; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 74; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - ((1008.39 / 6.0221409e+23) \* 5.11221307704105e-7 - ((36.888 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - (1845.89 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2853.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3893.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4971 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7179 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8453 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19010 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21310 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23830 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5068.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10137.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 40548.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 42238 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3.0590465 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function xenon\_unbound\_() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 54; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 78; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - ((1170.35 / 6.0221409e+23) \* 5.11221307704105e-7 - ((36.888 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - (2023.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2996 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4072 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6436 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8838 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10230 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17350 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19490 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (59400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2534.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2741.7 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5483.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10966.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 43867.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45695 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_33\_caesium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 55; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 78; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (51.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((375.7 / 6.0221409e+23) \* 5.11221307704105e-7 - ((32.21 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (51.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (2234.35 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3202.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4150 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6670 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10620 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20580 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22480 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25180 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (57600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (60700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2830.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5661.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11322.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45291.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47179 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_65\_barium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 56; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 82; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (317.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((502.85 / 6.0221409e+23) \* 5.11221307704105e-7 - ((28.07 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (317.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (965.22 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3458.4 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4530 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6850 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2861.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5722.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11445.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 45782.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 47690 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_51\_lanthanum() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 57; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 82; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (553.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((538.09 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.11 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (553.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1079.18 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1850.33 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4820 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5940 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8490 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11480 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16280 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26530 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2009.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3013.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6027.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12054.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 48218.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 50228 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_53\_cerium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 58; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 82; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (549.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((534.39 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.94 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (549.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1057.09 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1948.75 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3560.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6320 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7490 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8780 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10230 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12060 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13510 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2071.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2071.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3107.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6215.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12430.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 49722.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51794 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_28\_europium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 63; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 90; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (311.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((547.11 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.66 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (311.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1084.5 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (60800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2401.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3601.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7203.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14406.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 57627.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 60029 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_39\_samarium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 62; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 90; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (373.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((544.54 / 6.0221409e+23) \* 5.11221307704105e-7 - ((29.54 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (373.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1068.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2270 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4020 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6050 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8390 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9940 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (59500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2373.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3559.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7119.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 14238.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 56954.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 59328 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_40\_praseodimium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 59; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 82; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (507.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((527.79 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.2 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (507.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1025.7 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2086.37 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3761.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7910 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9360 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (56900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2135.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2135.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2135.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3203.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6406.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12812.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 51250.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 53386 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_52\_neodymium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 60; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 82; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (457.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((533.08 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.45 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (457.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1040.4 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2130 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3920 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5790 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2200.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2200.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2200.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2200.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3300.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6600.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 13201.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 52805.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 55006 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_21\_terbium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 65; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 94; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (461.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((565.77 / 6.0221409e+23) \* 5.11221307704105e-7 - ((28.91 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (461.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1110.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2110 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3790 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6420 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8680 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (59100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (62800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2620.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3930.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7861.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15723.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 62893.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 65514 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_22\_holmium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 67; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 98; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (335.5292 / 6.0221409e+23) \* 5.11221307704105e-7) - ((580.99 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.15 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (335.5292 / 6.0221409e+23) \* 5.11221307704105e-7) - (1136.7 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6170 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9170 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2764.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4146.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8292.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16585.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 66343.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 69108 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_37\_dysprosium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 66; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 98; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (365.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((573.03 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.7 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (365.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1123.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2210 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5990 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8970 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (64100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2691.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4037.82 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8075.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 16151.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 64605.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 67297 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_38\_gadolinium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count7 = 0; count7 < 64; count7++) {

hadronlist.push('proton');

}

for (var count8 = 0; count8 < 94; count8++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (470.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((593.37 / 6.0221409e+23) \* 5.11221307704105e-7 - ((37.03 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (470.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1165.2 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4290 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6250 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (58000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (61700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2550.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3825.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7650.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15301.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 61207.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 63758 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_18\_thulium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 69; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 100; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (311.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((596.7 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.03 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (311.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1164.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2280 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4090 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6310 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2912.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4368.66 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8737.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17474.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 69898.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72811 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_35\_ytterbium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 70; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 104; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - ((603.43 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.74 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - (1175.11 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2417.2 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4210 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6330 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9550 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3028.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4542.18 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9084.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18168.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 72674.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 75703 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_36\_erbium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 68; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 98; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (365.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((589.3 / 6.0221409e+23) \* 5.11221307704105e-7 - ((28.12 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (365.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1149.7 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2190 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4090 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6280 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9260 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (30000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2837.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4256.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 8513.52 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 17027.04 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 68108.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 70946 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_19\_lutetium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 71; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 104; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - ((523.52 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - (1363 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2022.27 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4365.9 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6450 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3104.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4657.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9314.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 18629.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 74517.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 77622 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_26\_tantalum() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 73; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 108; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (559.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((728.42 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.73 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (559.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1560 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2230 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3380 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4657.5 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9070 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3261.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4892.58 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9785.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19570.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 78281.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 81543 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_34\_hafnium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 72; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 108; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - ((658.52 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.73 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (152.55 / 6.0221409e+23) \* 5.11221307704105e-7) - (1410 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2176 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3220 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3182.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4774.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 9548.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 19096.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 76386.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 79569 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_1\_rhenium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 75; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 110; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = 0 - ((755.82 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.48 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = 0 - (1600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2610 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5010 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3422.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5134.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10268.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20537.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 82149.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 85572 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_24\_tungsten() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 74; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 110; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (407.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((758.76 / 6.0221409e+23) \* 5.11221307704105e-7 - ((24.27 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (407.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1580 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2510 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3690 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4979 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6249 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11770 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13620 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15460 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17270 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20160 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22350 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (52400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3341.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5012.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10025.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 20050.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 80202.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 83544 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_6\_osmium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 76; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 116; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (55.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((814.17 / 6.0221409e+23) \* 5.11221307704105e-7 - ((24.7 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (55.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2410 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3960 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5310 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6760 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8210 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9840 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2920.93333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3505.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5257.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10515.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21030.72 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 84122.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 87628 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_3\_iridium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 77; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 116; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (106.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((865.19 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.1 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (106.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3860 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6950 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (49200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2990.33333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 2990.33333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3588.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5382.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 10765.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 21530.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 86121.6 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 89710 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_7\_gold() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 79; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 118; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (110.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((890.13 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.418 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (110.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1949.3 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2890 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4340 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5790 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7140 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9070 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (47000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3131.93333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3131.93333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3131.93333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3131.93333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3758.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5637.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11274.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22549.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 90199.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 93958 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_8\_platinum() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 78; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 117; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (101.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((864.4 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.86 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (101.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1791 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4150 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7240 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8780 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (10500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (28300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3060.66666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3060.66666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3060.66666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3672.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5509.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11018.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 22036.8 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 88147.2 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 91820 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_13\_mercury() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 80; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 122; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (8.6 / 6.0221409e+23) \* 5.11221307704105e-7) - ((1007.07 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.983 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (8.6 / 6.0221409e+23) \* 5.11221307704105e-7) - (1809.76 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3325 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4684 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5905 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7390 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8970 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (14800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (18600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (51100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (54000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3237.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3237.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3237.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3237.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3237.4 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3884.88 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5827.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11654.64 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23309.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 93237.12 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97122 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_20\_thallium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 81; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 124; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (31.5 / 6.0221409e+23) \* 5.11221307704105e-7) - ((589.36 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.32 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (31.5 / 6.0221409e+23) \* 5.11221307704105e-7) - (1971.03 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2880.28 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4934 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6040 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7720 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9450 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (45100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3310.46666667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3972.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5958.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 11917.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 23835.36 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 95341.44 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99314 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_43\_lead() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 82; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 126; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (132.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((715.6 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.65 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (132.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1450.42 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3081.48 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4084.47 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6640 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9660 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (19600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (23600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (55000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3384.43333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4061.32 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6091.98 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12183.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24367.92 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 97471.68 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 101533 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_31\_uranium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count = 0; count < 92; count++) {

hadronlist.push('proton');

}

for (var count2 = 0; count2 < 146; count2++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (474.15 / 6.0221409e+23) \* 5.11221307704105e-7) - ((597.63 / 6.0221409e+23) \* 5.11221307704105e-7 - ((27.665 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (474.15 / 6.0221409e+23) \* 5.11221307704105e-7) - (1120 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1910 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (3540 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4440 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5980 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8590 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9750 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (16700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (21900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (31200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (33600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (36200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (38800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (41600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (48000 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4240.63333333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5088.76 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7633.14 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15266.28 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 30532.56 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 122130.24 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 127219 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_41\_thorium() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count3 = 0; count3 < 90; count3++) {

hadronlist.push('proton');

}

for (var count4 = 0; count4 < 142; count4++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (608.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((608.5 / 6.0221409e+23) \* 5.11221307704105e-7 - ((26.23 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (608.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1167 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (1768 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2764.1 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5600 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (6670 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (7910 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9170 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (12800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (25300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (27500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (29900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (32400 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (34900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (37500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (40900 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (43500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (46300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4422.16435185 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 5306.59722222 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 7959.89583333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 15919.7916667 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 31839.5833333 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 122264 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 127358.333333 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function my\_9\_bismuth() {

atx = cursor1x;

aty = cursor1y;

atz = cursor1z;

hadronlist = [];

for (var count5 = 0; count5 < 83; count5++) {

hadronlist.push('proton');

}

for (var count6 = 0; count6 < 126; count6++) {

hadronlist.push('neutron');

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz + 6.187927353e+22;

energy = (0 - (97.65 / 6.0221409e+23) \* 5.11221307704105e-7) - ((702.95 / 6.0221409e+23) \* 5.11221307704105e-7 - ((25.52 / 6.0221409e+23) \* 5.11221307704105e-10) \* material\_temperature\_\_kelvins);

add\_electron();

cursor1z = atz - 6.187927353e+22;

energy = (0 - (97.65 / 6.0221409e+23) \* 5.11221307704105e-7) - (1611.6 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

if (unused\_variable == 5) {

cursor1x = atx;

cursor1z = atz + 1e+22;

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (2466.5 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (4378 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (5292.8 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (8530 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (9940 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (11800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (13800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (15500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (17700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (20100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (22100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (24300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (26300 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (35700 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (39500 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (42100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (44800 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (50200 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - (53100 / 6.0221409e+23) \* 5.11221307704105e-7;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 3459.3 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 4151.16 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 6226.74 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 12453.48 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 24906.96 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 99627.84 \* 8.190671898165e-29;

add\_electron();

cursor1x = cursor1x + 6187927353732900000;

energy = 0 - 103779 \* 8.190671898165e-29;

add\_electron();

}

cursor1x = atx;

cursor1y = aty;

cursor1z = atz;

}

function obtain\_string\_location() {

string\_x = (string\_solar\_distance\_\_\_selected\_[0])[0];

string\_y = (string\_solar\_distance\_\_\_selected\_[0])[1];

string\_z = (string\_solar\_distance\_\_\_selected\_[0])[2];

}

function physical\_change() {

universe = 1;

while (universe <= list\_universes.length) {

allparticles = (list\_universes[universe - 1])[1];

var repeat\_end5 = 1.855094832e+43;

for (var count5 = 0; count5 < repeat\_end5; count5++) {

var repeat\_end4 = (list\_universes[universe - 1])[0];

for (var count4 = 0; count4 < repeat\_end4; count4++) {

var repeat\_end3 = 1e+100;

for (var count3 = 0; count3 < repeat\_end3; count3++) {

update\_universe();

}

}

}

universe = universe + 1;

}

}

function update\_universe() {

fluctuate\_vacuum();

particle1 = 1;

while (particle1 <= allparticles.length) {

selstring1 = allparticles[particle1 - 1];

update\_string();

particle1 = particle1 + 1;

}

}

function update\_string() {

correct\_energy();

if (viable == true) {

limit\_location();

rotate\_string();

move\_string();

limit\_string\_density();

vibrate\_string();

apply\_forces();

attempt\_breakage();

if (viable == true) {

attempt\_recombination();

}

}

}

function fluctuate\_vacuum() {

for (var count = 0; count < 10; count++) {

vcen1 = [math\_random\_int(1, 5.500001e+61 - 1) + math\_random\_int(1, 1e+100) / 1e+100, math\_random\_int(1, 5.500001e+61 - 1) + math\_random\_int(1, 1e+100) / 1e+100, math\_random\_int(1, 5.500001e+61 - 1) + math\_random\_int(1, 1e+100) / 1e+100];

randomizer = math\_random\_int(1, 1e+200) / 1e+200;

if (randomizer <= 2.4838955e+92 / (10 \* (1e+100 \* (1 / (0.0141347902 \* 8.19067189816475e-29))))) {

energy = 8.19067189816475e-29 \* (math\_random\_int(1, 5.500001e+71) / 5.500001e+61);

innitiate\_vacuum\_fluctuation();

}

}

}

function innitiate\_vacuum\_fluctuation() {

particle1 = 1;

viable2 = true;

while (particle1 <= allparticles.length) {

selstring1 = allparticles[particle1 - 1];

if (Math.sqrt(Math.abs(Math.pow(Math.abs(vcen1[2] - (selstring1[0])[2]), 2) + (Math.pow(Math.abs(vcen1[1] - (selstring1[0])[1]), 2) + Math.pow(Math.abs(vcen1[0] - (selstring1[0])[0]), 2)))) < 1 / energy && (selstring1[2])[1] >= 2) {

viable2 = false;

}

if ((vcen1[0] < 1 + 1 / energy || vcen1[0] > 5.500001e+61 - 1 / energy) || ((vcen1[1] < 1 + 1 / energy || vcen1[1] > 5.500001e+61 - 1 / energy) || (vcen1[2] < 1 + 1 / energy || vcen1[2] > 5.500001e+61 - 1 / energy))) {

viable2 = false;

}

particle1 = particle1 + 1;

}

if (viable2 == true) {

srho = 1 / energy - 1;

seta = (2 \* Math.PI) \* (math\_random\_int(1, 1e+100) / 1e+100);

sphi = Math.PI \* (math\_random\_int(1, 1e+100) / 1e+100);

vcen2 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [vcen2[0] + vcen1[0], vcen2[1] + vcen1[1], vcen2[2] + vcen1[2]];

vacuum\_fluctuation\_1();

vcen2 = [srho \* (Math.sin((Math.PI - sphi)) \* Math.cos((1 \* Math.PI + seta))), srho \* (Math.sin((Math.PI - sphi)) \* Math.sin((1 \* Math.PI + seta))), srho \* Math.cos((Math.PI - sphi))];

vcen2 = [vcen2[0] + vcen1[0], vcen2[1] + vcen1[1], vcen2[2] + vcen1[2]];

vacuum\_fluctuation\_2();

}

}

function vacuum\_fluctuation\_1() {

randomizer = math\_random\_int(1, 1e+100);

allparticles.push([[vcen2[0], vcen2[1], vcen2[2]], [1 \* Math.PI + seta, Math.PI - sphi, 1], ['open', energy, 1, 0, 0, true], [[], [], []]]);

if (randomizer <= 5e+99) {

((allparticles.slice(-1)[0])[2])[5] = false;

}

cursor5x = vcen2[0] - energy / 2;

cursor5y = vcen2[1];

cursor5z = vcen2[2];

var repeat\_end3 = Math.ceil(energy \* 1e+100);

for (var count3 = 0; count3 < repeat\_end3; count3++) {

cursor5x = cursor5x + 1e-100;

(((allparticles.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((allparticles.slice(-1)[0])[3])[1] = ((allparticles.slice(-1)[0])[3])[0];

var repeat\_end4 = Math.ceil(energy \* 1e+100);

for (var count4 = 0; count4 < repeat\_end4; count4++) {

(((allparticles.slice(-1)[0])[3])[2]).push(0);

}

}

function vacuum\_fluctuation\_2() {

allparticles.push([[vcen2[0], vcen2[1], vcen2[2]], [seta, sphi, 1], ['open', 0 - energy, 1, 0, 0, false], [[], [], []]]);

if (randomizer <= 5e+99) {

((allparticles.slice(-1)[0])[2])[5] = true;

}

cursor5x = vcen2[0] - energy / 2;

cursor5y = vcen2[1];

cursor5z = vcen2[2];

var repeat\_end5 = Math.ceil(energy \* 1e+100);

for (var count5 = 0; count5 < repeat\_end5; count5++) {

cursor5x = cursor5x + 1e-100;

(((allparticles.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((allparticles.slice(-1)[0])[3])[1] = ((allparticles.slice(-1)[0])[3])[0];

var repeat\_end6 = Math.ceil(energy \* 1e+100);

for (var count6 = 0; count6 < repeat\_end6; count6++) {

(((allparticles.slice(-1)[0])[3])[2]).push(0);

}

}

function add\_electron() {

energy = 510998.95 \* 8.19067189816475e-29 + energy;

movex = atom\_speed\_x;

movey = atom\_speed\_y;

movez = atom\_speed\_z;

if (Math.abs(movex) < 1e-100) {

movex = 1e-100;

}

if (Math.abs(movey) < 1e-100) {

movey = 1e-100;

}

if (Math.abs(movez) < 1e-100) {

movez = 1e-100;

}

seta = Math.atan(movey / movex);

if (movey < 0) {

seta = Math.PI + (Math.PI + seta);

}

sphi = Math.acos(movez / Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2)))));

srho = Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2))));

energy = energy + energy \* (1 / Math.sqrt(Math.abs(1 - Math.pow(srho, 2) / Math.pow(1, 2))));

list\_\_universe\_string\_entities\_\_\_default.push([[cursor1x, cursor1y, cursor1z], [seta, sphi, srho], ['open', energy, 2, 0, 1, true], [[], [], []]]);

cursor5x = cursor1x - energy / 2;

cursor5y = cursor1y;

cursor5z = cursor1z;

var repeat\_end = Math.ceil(energy \* 1e+100);

for (var count = 0; count < repeat\_end; count++) {

cursor5x = cursor5x + 1e-100;

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[1] = ((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0];

var repeat\_end2 = Math.ceil(energy \* 1e+100);

for (var count2 = 0; count2 < repeat\_end2; count2++) {

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[2]).push(0);

}

}

function add\_randomized\_solar\_photon() {

list\_\_universe\_string\_entities\_\_\_default.push([[cursor1x, cursor1y, cursor1z], [math\_random\_int(1, (2 \* Math.PI) \* 1e+30) / 1e+30, math\_random\_int(1, Math.PI \* 1e+30) / 1e+30, 1], ['open', 2.137673 \* 8.1907e-29, 1, 0, 0, true], [[], [], []]]);

cursor5x = cursor1x - (2.137673 \* 8.1907e-29) / 2;

cursor5y = cursor1y;

cursor5z = cursor1z;

var repeat\_end3 = Math.ceil((2.137673 \* 8.1907e-29) \* 1e+100);

for (var count3 = 0; count3 < repeat\_end3; count3++) {

cursor5x = cursor5x + 1e-100;

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[1] = ((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0];

var repeat\_end4 = Math.ceil((2.137673 \* 8.1907e-29) \* 1e+100);

for (var count4 = 0; count4 < repeat\_end4; count4++) {

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[2]).push(0);

}

}

function adddownquark() {

if (hadronbonus == 1) {

energy = ((1.00727646688 \* 931494102.42) / 3 + (1.00866491578 - 1.00727646688) \* 931494102.42) \* 8.19067189816475e-29;

} else {

energy = ((1.00727646688 \* 931494102.42) / 3) \* 8.19067189816475e-29;

}

movex = atom\_speed\_x;

movey = atom\_speed\_y;

movez = atom\_speed\_z;

if (Math.abs(movex) < 1e-100) {

movex = 1e-100;

}

if (Math.abs(movey) < 1e-100) {

movey = 1e-100;

}

if (Math.abs(movez) < 1e-100) {

movez = 1e-100;

}

seta = Math.atan(movey / movex);

if (movey < 0) {

seta = Math.PI + (Math.PI + seta);

}

sphi = Math.acos(movez / Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2)))));

srho = Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2))));

energy = energy + energy \* (1 / Math.sqrt(Math.abs(1 - Math.pow(srho, 2) / Math.pow(1, 2))));

list\_\_universe\_string\_entities\_\_\_default.push([[cursor1x, cursor1y, cursor1z], [seta, sphi, srho], ['open', energy, 2, 0, 1, true], [[], [], []]]);

cursor5x = cursor1x - energy / 2;

cursor5y = cursor1y;

cursor5z = cursor1z;

var repeat\_end = Math.ceil(energy \* 1e+100);

for (var count = 0; count < repeat\_end; count++) {

cursor5x = cursor5x + 1e-100;

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[1] = ((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0];

var repeat\_end2 = Math.ceil(energy \* 1e+100);

for (var count2 = 0; count2 < repeat\_end2; count2++) {

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[2]).push(0);

}

}

function addupquark() {

energy = ((1.00727646688 \* 931494102.42) / 3) \* 8.19067189816475e-29;

movex = atom\_speed\_x;

movey = atom\_speed\_y;

movez = atom\_speed\_z;

if (Math.abs(movex) < 1e-100) {

movex = 1e-100;

}

if (Math.abs(movey) < 1e-100) {

movey = 1e-100;

}

if (Math.abs(movez) < 1e-100) {

movez = 1e-100;

}

seta = Math.atan(movey / movex);

if (movey < 0) {

seta = Math.PI + (Math.PI + seta);

}

sphi = Math.acos(movez / Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2)))));

srho = Math.sqrt(Math.abs(Math.pow(Math.abs(movex), 2) + (Math.pow(Math.abs(movey), 2) + Math.pow(Math.abs(movez), 2))));

energy = energy + energy \* (1 / Math.sqrt(Math.abs(1 - Math.pow(srho, 2) / Math.pow(1, 2))));

list\_\_universe\_string\_entities\_\_\_default.push([[cursor1x, cursor1y, cursor1z], [seta, sphi, srho], ['open', energy, 2, 0, 2, true], [[], [], []]]);

cursor5x = cursor1x - energy / 2;

cursor5y = cursor1y;

cursor5z = cursor1z;

var repeat\_end3 = Math.ceil(energy \* 1e+100);

for (var count3 = 0; count3 < repeat\_end3; count3++) {

cursor5x = cursor5x + 1e-100;

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0]).push([cursor5x, cursor5y, cursor5z, 0, 0, 0, 0, 0, 0]);

}

((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[1] = ((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[0];

var repeat\_end4 = Math.ceil(energy \* 1e+100);

for (var count4 = 0; count4 < repeat\_end4; count4++) {

(((list\_\_universe\_string\_entities\_\_\_default.slice(-1)[0])[3])[2]).push(0);

}

}

function limit\_location() {

string\_solar\_distance\_\_\_selected\_ = selstring1;

obtain\_string\_location();

if (string\_x < 1) {

(selstring1[0])[0] = (selstring1[0])[0] + 5.500001e+61;

(selstring1[0])[0] = (selstring1[0])[0] - 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - 2;

nullbrane1 = nullbrane1 + 1;

}

} else if (string\_x > 5.500001e+61) {

(selstring1[0])[0] = (selstring1[0])[0] - 5.500001e+61;

(selstring1[0])[0] = (selstring1[0])[0] + 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + 2;

nullbrane1 = nullbrane1 + 1;

}

}

if (string\_y < 1) {

(selstring1[0])[1] = (selstring1[0])[1] + 5.500001e+61;

(selstring1[0])[1] = (selstring1[0])[1] - 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - 2;

nullbrane1 = nullbrane1 + 1;

}

} else if (string\_y > 5.500001e+61) {

(selstring1[0])[1] = (selstring1[0])[1] - 5.500001e+61;

(selstring1[0])[1] = (selstring1[0])[1] + 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + 2;

nullbrane1 = nullbrane1 + 1;

}

}

if (string\_z < 1) {

(selstring1[0])[2] = (selstring1[0])[2] + 5.500001e+61;

(selstring1[0])[2] = (selstring1[0])[2] - 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - 2;

nullbrane1 = nullbrane1 + 1;

}

} else if (string\_z > 5.500001e+61) {

(selstring1[0])[2] = (selstring1[0])[2] - 5.500001e+61;

(selstring1[0])[2] = (selstring1[0])[2] + 2;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - 5.500001e+61;

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + 2;

nullbrane1 = nullbrane1 + 1;

}

}

}

function correct\_energy() {

viable = true;

if (((allparticles[particle1 - 1])[2])[1] == 0) {

allparticles.splice(particle1 - 1, 1);

viable = false;

particle1 = particle1 - 1;

} else if (((allparticles[particle1 - 1])[3])[0].length > Math.ceil(Math.abs(((allparticles[particle1 - 1])[2])[1]) \* 1e+100)) {

while (((allparticles[particle1 - 1])[3])[0].length > Math.ceil(Math.abs(((allparticles[particle1 - 1])[2])[1]) \* 1e+100)) {

(((allparticles[particle1 - 1])[3])[0]).pop();

(((allparticles[particle1 - 1])[3])[2]).pop();

}

((allparticles[particle1 - 1])[3])[1] = ((allparticles[particle1 - 1])[3])[0];

} else if (((allparticles[particle1 - 1])[3])[0].length < Math.ceil(Math.abs(((allparticles[particle1 - 1])[2])[1]) \* 1e+100)) {

while (((allparticles[particle1 - 1])[3])[0].length < Math.ceil(Math.abs(((allparticles[particle1 - 1])[2])[1]) \* 1e+100)) {

(((allparticles[particle1 - 1])[3])[0]).push((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0]);

(((allparticles[particle1 - 1])[3])[2]).push(0);

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[2] = ((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[2] + 1e-110;

}

((allparticles[particle1 - 1])[3])[1] = ((allparticles[particle1 - 1])[3])[0];

}

if (viable == true && ((allparticles[particle1 - 1])[2])[0] == 'closed') {

var tmp\_list = (((allparticles[particle1 - 1])[3])[0]);

tmp\_list[tmp\_list.length - 1] = (((allparticles[particle1 - 1])[3])[0])[0];

((allparticles[particle1 - 1])[3])[1] = ((allparticles[particle1 - 1])[3])[0];

}

if (viable == true && ((allparticles[particle1 - 1])[2])[0] == 'open') {

((((allparticles[particle1 - 1])[3])[0])[0])[3] = 0;

((((allparticles[particle1 - 1])[3])[0])[0])[4] = 0;

((((allparticles[particle1 - 1])[3])[0])[0])[5] = 0;

((((allparticles[particle1 - 1])[3])[0])[0])[6] = 0;

((((allparticles[particle1 - 1])[3])[0])[0])[7] = 0;

((((allparticles[particle1 - 1])[3])[0])[0])[8] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[3] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[4] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[5] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[6] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[7] = 0;

((((allparticles[particle1 - 1])[3])[0]).slice(-1)[0])[8] = 0;

((allparticles[particle1 - 1])[3])[1] = ((allparticles[particle1 - 1])[3])[0];

}

}

function limit\_string\_density() {

if ((selstring1[3])[0].length > 1) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

if (((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0])[0]) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8]), 2)];

dist2 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist2 == 0) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + 1e-110;

} else if (dist2 > 1e-100) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

} else if (dist2 < 1e-100) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

}

} else if (((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0]).slice(-1)[0]) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8]), 2)];

dist1 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist1 == 0) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + 1e-110;

} else if (dist1 > 1e-100) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

} else if (dist1 < 1e-100) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

}

} else {

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8]), 2)];

dist1 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8]), 2)];

dist2 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist1 == 0 || dist2 == 0) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + 1e-110;

} else if (dist1 > 1e-100 || dist2 > 1e-100) {

if (dist1 > dist2) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

} else {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

}

} else if (dist1 < 1e-100 || dist2 < 1e-100) {

if (dist1 < dist2) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

} else {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

if ((((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] < (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[largest2 - 1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] - dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] = (((selstring1[3])[0])[nullbrane1 - 1])[largest2 - 1] + dist3[largest2 - 1] \* (0.134567 + Math.PI / 150);

}

}

}

}

nullbrane1 = nullbrane1 + 1;

}

}

}

function move\_string() {

movex = (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0]));

movey = (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0]));

movez = (selstring1[1])[2] \* Math.cos((selstring1[1])[1]);

movex = movex / 1e+100;

movey = movey / 1e+100;

movez = movez / 1e+100;

if (unused\_variable == 5) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + movex;

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[0] = 0;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(selstring1[0])[0] = (selstring1[0])[0] + (((selstring1[3])[0])[nullbrane1 - 1])[0];

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[0] = (selstring1[0])[0] / (selstring1[3])[0].length;

}

if (unused\_variable == 5) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + movey;

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[1] = 0;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(selstring1[0])[1] = (selstring1[0])[1] + (((selstring1[3])[0])[nullbrane1 - 1])[1];

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[1] = (selstring1[0])[1] / (selstring1[3])[0].length;

}

if (unused\_variable == 5) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + movez;

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[2] = 0;

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

(selstring1[0])[2] = (selstring1[0])[2] + (((selstring1[3])[0])[nullbrane1 - 1])[2];

nullbrane1 = nullbrane1 + 1;

}

(selstring1[0])[2] = (selstring1[0])[2] / (selstring1[3])[0].length;

}

}

function apply\_forces() {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

vloc = [0, 0, 0, 0, 0, 0, 0, 0, 0];

if (unused\_variable == 5) {

particle2 = 1;

while (particle2 <= allparticles.length) {

if (unused\_variable == 5) {

selstring2 = allparticles[particle2 - 1];

nullbrane2 = 1;

while (nullbrane2 <= (selstring2[3])[0].length) {

if (unused\_variable == 5) {

dist3 = [(((selstring2[3])[0])[nullbrane2 - 1])[0] - (((selstring1[3])[0])[nullbrane1 - 1])[0], (((selstring2[3])[0])[nullbrane2 - 1])[1] - (((selstring1[3])[0])[nullbrane1 - 1])[1], (((selstring2[3])[0])[nullbrane2 - 1])[2] - (((selstring1[3])[0])[nullbrane1 - 1])[2], (((selstring2[3])[0])[nullbrane2 - 1])[3] - (((selstring1[3])[0])[nullbrane1 - 1])[3], (((selstring2[3])[0])[nullbrane2 - 1])[4] - (((selstring1[3])[0])[nullbrane1 - 1])[4], (((selstring2[3])[0])[nullbrane2 - 1])[5] - (((selstring1[3])[0])[nullbrane1 - 1])[5], (((selstring2[3])[0])[nullbrane2 - 1])[6] - (((selstring1[3])[0])[nullbrane1 - 1])[6], (((selstring2[3])[0])[nullbrane2 - 1])[7] - (((selstring1[3])[0])[nullbrane1 - 1])[7], (((selstring2[3])[0])[nullbrane2 - 1])[8] - (((selstring1[3])[0])[nullbrane1 - 1])[8]];

dist5 = [Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[0] - (((selstring1[3])[0])[nullbrane1 - 1])[0]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[1] - (((selstring1[3])[0])[nullbrane1 - 1])[1]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[2] - (((selstring1[3])[0])[nullbrane1 - 1])[2]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[3] - (((selstring1[3])[0])[nullbrane1 - 1])[3]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[4] - (((selstring1[3])[0])[nullbrane1 - 1])[4]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[5] - (((selstring1[3])[0])[nullbrane1 - 1])[5]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[6] - (((selstring1[3])[0])[nullbrane1 - 1])[6]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[7] - (((selstring1[3])[0])[nullbrane1 - 1])[7]), 2), Math.pow(Math.abs((((selstring2[3])[0])[nullbrane2 - 1])[8] - (((selstring1[3])[0])[nullbrane1 - 1])[8]), 2)];

dist4 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist4 < 1) {

dist3[0] = dist3[0] / (dist4 / Math.pow(dist4, 2));

dist3[1] = dist3[1] / (dist4 / Math.pow(dist4, 2));

dist3[2] = dist3[2] / (dist4 / Math.pow(dist4, 2));

dist3[3] = dist3[3] / (dist4 / Math.pow(dist4, 2));

dist3[4] = dist3[4] / (dist4 / Math.pow(dist4, 2));

dist3[5] = dist3[5] / (dist4 / Math.pow(dist4, 2));

dist3[6] = dist3[6] / (dist4 / Math.pow(dist4, 2));

dist3[7] = dist3[7] / (dist4 / Math.pow(dist4, 2));

dist3[8] = dist3[8] / (dist4 / Math.pow(dist4, 2));

} else {

dist3[0] = dist3[0] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[1] = dist3[1] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[2] = dist3[2] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[3] = dist3[3] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[4] = dist3[4] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[5] = dist3[5] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[6] = dist3[6] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[7] = dist3[7] / (dist4 / (1 / Math.pow(dist4, 2)));

dist3[8] = dist3[8] / (dist4 / (1 / Math.pow(dist4, 2)));

}

vloc[0] = vloc[0] + dist3[0];

vloc[1] = vloc[1] + dist3[1];

vloc[2] = vloc[2] + dist3[2];

vloc[3] = vloc[3] + dist3[3];

vloc[4] = vloc[4] + dist3[4];

vloc[5] = vloc[5] + dist3[5];

vloc[6] = vloc[6] + dist3[6];

vloc[7] = vloc[7] + dist3[7];

vloc[8] = vloc[8] + dist3[8];

}

nullbrane2 = nullbrane2 + 1;

}

}

particle2 = particle2 + 1;

}

}

if (vloc[0] == Infinity) {

vloc[0] = 1e+100;

}

if (vloc[1] == Infinity) {

vloc[1] = 1e+100;

}

if (vloc[2] == Infinity) {

vloc[2] = 1e+100;

}

if (vloc[3] == Infinity) {

vloc[3] = 1e+100;

}

if (vloc[4] == Infinity) {

vloc[4] = 1e+100;

}

if (vloc[5] == Infinity) {

vloc[5] = 1e+100;

}

if (vloc[6] == Infinity) {

vloc[6] = 1e+100;

}

if (vloc[7] == Infinity) {

vloc[7] = 1e+100;

}

if (vloc[8] == Infinity) {

vloc[8] = 1e+100;

}

dist1 = [Math.pow(vloc[0] / 1e+100, 2), Math.pow(vloc[1] / 1e+100, 2), Math.pow(vloc[2] / 1e+100, 2), Math.pow(vloc[3] / 1e+100, 2), Math.pow(vloc[4] / 1e+100, 2), Math.pow(vloc[5] / 1e+100, 2), Math.pow(vloc[6] / 1e+100, 2), Math.pow(vloc[7] / 1e+100, 2), Math.pow(vloc[8] / 1e+100, 2)];

dist2 = Math.sqrt(dist1.reduce(function(x, y) {return x + y;})) / 1e+100;

if (dist2 == Infinity) {

dist2 = 1e-100;

}

vloc[0] = vloc[0] / 1e+200;

vloc[1] = vloc[1] / 1e+200;

vloc[2] = vloc[2] / 1e+200;

vloc[3] = vloc[3] / 1e+200;

vloc[4] = vloc[4] / 1e+200;

vloc[5] = vloc[5] / 1e+200;

vloc[6] = vloc[6] / 1e+200;

vloc[7] = vloc[7] / 1e+200;

vloc[8] = vloc[8] / 1e+200;

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + vloc[0];

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + vloc[1];

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + vloc[2];

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] + vloc[3];

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] + vloc[4];

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] + vloc[5];

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] + vloc[6];

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] + vloc[7];

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] + vloc[8];

((selstring1[3])[2])[nullbrane1 - 1] = dist2;

nullbrane1 = nullbrane1 + 1;

}

}

function vibrate\_string() {

if ((selstring1[3])[0].length > 1) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

if ((selstring1[2])[0] == 'open') {

if (((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0]).slice(-1)[0]) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

} else {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

}

wavechange = 2 \* (1e-100 / (selstring1[2])[1]);

if ((selstring1[2])[3] >= 0 && (selstring1[2])[3] < 1) {

if (unused\_variable == true) {

if (largest2 != 1) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 2) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 3) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

}

} else if ((selstring1[2])[3] >= 1 && (selstring1[2])[3] < 2) {

if (unused\_variable == true) {

if (largest2 != 1) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 2) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 3) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

}

} else if ((selstring1[2])[3] >= 2 && (selstring1[2])[3] < 3) {

if (unused\_variable == true) {

if (largest2 != 1) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 2) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 3) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

}

} else {

if (unused\_variable == true) {

if (largest2 != 1) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 2) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 3) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((selstring1[2])[2] \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / ((selstring1[2])[2] \* Math.PI)));

}

}

}

(selstring1[2])[3] = (selstring1[2])[3] + wavechange;

if ((selstring1[2])[3] >= 4) {

(selstring1[2])[3] = (selstring1[2])[3] - 4;

}

} else {

vibrate\_closed\_string();

}

nullbrane1 = nullbrane1 + 1;

}

}

}

function vibrate\_closed\_string() {

if (((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0]).slice(-1)[0]) {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 - 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

} else {

dist3 = [Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[0]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[1]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[2]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[3]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[4]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[5]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[6]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[7]), Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[(nullbrane1 + 1) - 1])[8])];

largest1 = dist3[0];

largest2 = 1;

if (dist3[1] > largest1) {

largest1 = dist3[1];

largest2 = 2;

}

if (dist3[2] > largest1) {

largest1 = dist3[2];

largest2 = 3;

}

if (dist3[3] > largest1) {

largest1 = dist3[3];

largest2 = 4;

}

if (dist3[4] > largest1) {

largest1 = dist3[4];

largest2 = 5;

}

if (dist3[5] > largest1) {

largest1 = dist3[5];

largest2 = 6;

}

if (dist3[6] > largest1) {

largest1 = dist3[6];

largest2 = 7;

}

if (dist3[7] > largest1) {

largest1 = dist3[7];

largest2 = 8;

}

if (dist3[8] > largest1) {

largest1 = dist3[8];

largest2 = 9;

}

}

wavechange = 2 \* (1e-100 / (selstring1[2])[1]);

if ((selstring1[2])[3] >= 0 && (selstring1[2])[3] < 1) {

if (unused\_variable == true) {

if (largest2 != 1) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[0] < (selstring1[0])[0]) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 2) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[1] < (selstring1[0])[1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 3) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[2] < (selstring1[0])[2]) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

} else if ((selstring1[2])[3] >= 1 && (selstring1[2])[3] < 2) {

if (unused\_variable == true) {

if (largest2 != 1) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[0] < (selstring1[0])[0]) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 2) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[1] < (selstring1[0])[1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 3) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[2] < (selstring1[0])[2]) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

} else if ((selstring1[2])[3] >= 2 && (selstring1[2])[3] < 3) {

if (unused\_variable == true) {

if (largest2 != 1) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[0] < (selstring1[0])[0]) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 2) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[1] < (selstring1[0])[1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 3) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[2] < (selstring1[0])[2]) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

} else {

if (unused\_variable == true) {

if (largest2 != 1) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[0] < (selstring1[0])[0]) {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (((selstring1[3])[0])[nullbrane1 - 1])[0] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 2) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[1] < (selstring1[0])[1]) {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (((selstring1[3])[0])[nullbrane1 - 1])[1] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 3) {

if ((((selstring1[3])[0])[nullbrane1 - 1])[2] < (selstring1[0])[2]) {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] - ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

} else {

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (((selstring1[3])[0])[nullbrane1 - 1])[2] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

if (largest2 != 4) {

(((selstring1[3])[0])[nullbrane1 - 1])[3] = (((selstring1[3])[0])[nullbrane1 - 1])[3] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 5) {

(((selstring1[3])[0])[nullbrane1 - 1])[4] = (((selstring1[3])[0])[nullbrane1 - 1])[4] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 6) {

(((selstring1[3])[0])[nullbrane1 - 1])[5] = (((selstring1[3])[0])[nullbrane1 - 1])[5] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 7) {

(((selstring1[3])[0])[nullbrane1 - 1])[6] = (((selstring1[3])[0])[nullbrane1 - 1])[6] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 8) {

(((selstring1[3])[0])[nullbrane1 - 1])[7] = (((selstring1[3])[0])[nullbrane1 - 1])[7] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

if (largest2 != 9) {

(((selstring1[3])[0])[nullbrane1 - 1])[8] = (((selstring1[3])[0])[nullbrane1 - 1])[8] + ((Math.abs(1) \* Math.sin(((nullbrane1 / (selstring1[3])[0].length) \* ((((selstring1[2])[2] + 1) \* 2) \* Math.PI)))) / Math.sqrt(8)) \* (wavechange \* ((selstring1[2])[1] / (((selstring1[2])[2] \* Math.PI) \* 4)));

}

}

}

(selstring1[2])[3] = (selstring1[2])[3] + wavechange;

if ((selstring1[2])[3] >= 4) {

(selstring1[2])[3] = (selstring1[2])[3] - 4;

}

}

function rotate\_string() {

if ((selstring1[3])[0].length > 1) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length) {

if (unused\_variable == 5) {

spinrelx = (((selstring1[3])[0])[nullbrane1 - 1])[0] - (selstring1[0])[0];

spinrely = (((selstring1[3])[0])[nullbrane1 - 1])[1] - (selstring1[0])[1];

spinrelz = (((selstring1[3])[0])[nullbrane1 - 1])[2] - (selstring1[0])[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

spineta = Math.atan(spinrely / spinrelx);

if (spinrely < 0) {

spineta = Math.PI + (Math.PI + spineta);

}

spinphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

spinrho = Math.sqrt(Math.abs((Math.pow(Math.abs(spinrelx), 2) + Math.pow(Math.abs(spinrely), 2)) + Math.pow(Math.abs(spinrelz), 2)));

spincom = [Math.abs(Math.PI / 2 - (selstring1[1])[1]), 0, 0];

if ((selstring1[1])[0] > Math.PI) {

spincom[1] = Math.abs(Math.PI \* (3 / 2) - (selstring1[1])[0]) \* ((Math.PI / 2 - spincom[0]) / (Math.PI / 2));

spincom[2] = Math.abs(Math.PI / 2 - Math.abs(Math.PI \* (3 / 2) - (selstring1[1])[0])) \* ((Math.PI / 2 - spincom[0]) / (Math.PI / 2));

} else {

spincom[1] = Math.abs(Math.PI / 2 - (selstring1[1])[0]) \* ((Math.PI / 2 - spincom[0]) / (Math.PI / 2));

spincom[2] = Math.abs(Math.PI / 2 - Math.abs(Math.PI / 2 - (selstring1[1])[0])) \* ((Math.PI / 2 - spincom[0]) / (Math.PI / 2));

}

if ((selstring1[2])[0] == 'open') {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

}

}

} else {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[0] / spincom.reduce(function(x, y) {return x + y;}));

}

}

}

if (spineta < 0) {

spineta = Math.PI + (Math.PI + spineta);

} else if (spineta > 2 \* Math.PI) {

spineta = spineta - 2 \* Math.PI;

}

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (selstring1[0])[0] + spinrho \* (Math.sin(spinphi) \* Math.cos(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (selstring1[0])[1] + spinrho \* (Math.sin(spinphi) \* Math.sin(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (selstring1[0])[2] + spinrho \* Math.cos(spinphi);

}

if (unused\_variable == 5) {

spinrelx = (((selstring1[3])[0])[nullbrane1 - 1])[0] - (selstring1[0])[0];

spinrely = (((selstring1[3])[0])[nullbrane1 - 1])[1] - (selstring1[0])[1];

spinrelz = (((selstring1[3])[0])[nullbrane1 - 1])[2] - (selstring1[0])[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

spineta = Math.atan(spinrelz / spinrelx);

if (spinrely < 0) {

spineta = Math.PI + (Math.PI + spineta);

}

spinphi = Math.acos(spinrely / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

spinrho = Math.sqrt(Math.abs((Math.pow(Math.abs(spinrelx), 2) + Math.pow(Math.abs(spinrely), 2)) + Math.pow(Math.abs(spinrelz), 2)));

if ((selstring1[2])[0] == 'open') {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

}

}

} else {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[2] / spincom.reduce(function(x, y) {return x + y;}));

}

}

}

if (spineta < 0) {

spineta = Math.PI + (Math.PI + spineta);

} else if (spineta > 2 \* Math.PI) {

spineta = spineta - 2 \* Math.PI;

}

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (selstring1[0])[0] + spinrho \* (Math.sin(spinphi) \* Math.cos(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (selstring1[0])[2] + spinrho \* (Math.sin(spinphi) \* Math.sin(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (selstring1[0])[1] + spinrho \* Math.cos(spinphi);

}

if (unused\_variable == 5) {

spinrelx = (((selstring1[3])[0])[nullbrane1 - 1])[0] - (selstring1[0])[0];

spinrely = (((selstring1[3])[0])[nullbrane1 - 1])[1] - (selstring1[0])[1];

spinrelz = (((selstring1[3])[0])[nullbrane1 - 1])[2] - (selstring1[0])[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

spineta = Math.atan(spinrely / spinrelz);

if (spinrely < 0) {

spineta = Math.PI + (Math.PI + spineta);

}

spinphi = Math.acos(spinrelx / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

spinrho = Math.sqrt(Math.abs((Math.pow(Math.abs(spinrelx), 2) + Math.pow(Math.abs(spinrely), 2)) + Math.pow(Math.abs(spinrelz), 2)));

if ((selstring1[2])[0] == 'open') {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / (selstring1[2])[1]) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

}

}

} else {

if ((selstring1[1])[1] < Math.PI / 2) {

if ((selstring1[2])[5] == true) {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

}

} else {

if ((selstring1[2])[5] == true) {

spineta = spineta - ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

} else {

spineta = spineta + ((2 \* 1e-100) / ((selstring1[2])[1] / Math.PI)) \* (spincom[1] / spincom.reduce(function(x, y) {return x + y;}));

}

}

}

if (spineta < 0) {

spineta = Math.PI + (Math.PI + spineta);

} else if (spineta > 2 \* Math.PI) {

spineta = spineta - 2 \* Math.PI;

}

(((selstring1[3])[0])[nullbrane1 - 1])[2] = (selstring1[0])[2] + spinrho \* (Math.sin(spinphi) \* Math.cos(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[1] = (selstring1[0])[1] + spinrho \* (Math.sin(spinphi) \* Math.sin(spineta));

(((selstring1[3])[0])[nullbrane1 - 1])[0] = (selstring1[0])[0] + spinrho \* Math.cos(spinphi);

}

nullbrane1 = nullbrane1 + 1;

}

}

}

function attempt\_breakage() {

if ((selstring1[3])[0].length > 1) {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length && viable == true) {

randomizer = math\_random\_int(1, 1e+300) / 1e+300;

if (randomizer <= ((selstring1[3])[2])[nullbrane1 - 1]) {

if ((selstring1[2])[0] == 'closed') {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp1[2])[0] = 'open';

(vp1[3])[0] = [];

(vp1[3])[2] = [];

nullbrane2 = nullbrane1;

while (nullbrane2 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

nullbrane2 = nullbrane2 + 1;

}

nullbrane2 = 1;

while (nullbrane2 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

nullbrane2 = nullbrane2 + 1;

}

nullbrane2 = 1;

while (nullbrane2 <= (selstring1[3])[0].length) {

((vp1[3])[2]).push(0);

nullbrane2 = nullbrane2 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

allparticles[particle1 - 1] = vp1;

viable = false;

}

} else if ((selstring1[2])[2] == 1) {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp1[2])[4] = 1;

(vp2[2])[4] = 2;

(vp1[2])[2] = 2;

(vp2[2])[2] = 2;

if (nullbrane1 < (selstring1[3])[0].length / 2) {

(vp1[2])[5] = !(selstring1[2])[5];

(vp2[2])[5] = (selstring1[2])[5];

} else {

(vp2[2])[5] = !(selstring1[2])[5];

(vp1[2])[5] = (selstring1[2])[5];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane2 = 1;

while (nullbrane2 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp1[3])[2]).push(0);

nullbrane2 = nullbrane2 + 1;

}

nullbrane2 = (selstring1[3])[0].length;

while (nullbrane2 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp2[3])[2]).push(0);

nullbrane2 = nullbrane2 - 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1];

(vp2[2])[1] = (((selstring1[3])[0].length - (nullbrane1 - 1)) / (selstring1[3])[0].length) \* (selstring1[2])[1];

vcen1 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane2 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane2 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane2 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane2 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (nullbrane1 < (selstring1[3])[0].length / 2) {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp2[1])[0] = 1 \* Math.PI + seta;

(vp2[1])[1] = Math.PI - sphi;

(vp1[1])[2] = 1 - Math.abs((vp1[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp1[2])[1] == 0) {

(vp2[1])[2] = 0;

} else {

(vp2[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp1[2])[1] / (vp2[2])[1] + 1), 2)));

}

} else {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp1[1])[0] = 1 \* Math.PI + seta;

(vp1[1])[1] = Math.PI - sphi;

(vp2[1])[2] = 1 - Math.abs((vp2[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp2[2])[1] == 0) {

(vp1[1])[2] = 0;

} else {

(vp1[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp2[2])[1] / (vp1[2])[1] + 1), 2)));

}

}

allparticles.splice(particle1 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

particle1 = particle1 + 1;

}

} else if ((selstring1[2])[2] == 2) {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

if (nullbrane1 < (selstring1[3])[0].length / 2) {

(vp1[2])[5] = !(selstring1[2])[5];

(vp2[2])[5] = (selstring1[2])[5];

(vp1[2])[2] = 1;

(vp2[2])[2] = 2;

(vp1[2])[4] = 0;

(vp2[2])[4] = (vp2[2])[4];

} else {

(vp2[2])[5] = !(selstring1[2])[5];

(vp1[2])[5] = (selstring1[2])[5];

(vp1[2])[2] = 2;

(vp2[2])[2] = 1;

(vp2[2])[4] = 0;

(vp1[2])[4] = (vp1[2])[4];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane2 = 1;

while (nullbrane2 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp1[3])[2]).push(0);

nullbrane2 = nullbrane2 + 1;

}

nullbrane2 = (selstring1[3])[0].length;

while (nullbrane2 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp2[3])[2]).push(0);

nullbrane2 = nullbrane2 - 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1];

(vp2[2])[1] = (((selstring1[3])[0].length - (nullbrane1 - 1)) / (selstring1[3])[0].length) \* (selstring1[2])[1];

vcen1 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane2 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane2 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane2 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane2 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (nullbrane1 < (selstring1[3])[0].length / 2) {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp1[1])[2] = 1;

if ((vp1[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp1[2])[1] / (vp2[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp2[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp2[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp2[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp2[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp2[1])[0] = (selstring1[1])[0];

(vp2[1])[1] = (selstring1[1])[1];

(vp2[1])[2] = (selstring1[1])[2];

}

} else {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp2[1])[2] = 1;

if ((vp2[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp2[2])[1] / (vp1[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp1[1])[0] = (selstring1[1])[0];

(vp1[1])[1] = (selstring1[1])[1];

(vp1[1])[2] = (selstring1[1])[2];

}

}

allparticles.splice(particle1 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

particle1 = particle1 + 1;

}

} else {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp1[2])[4] = 1;

(vp2[2])[4] = 2;

if (nullbrane1 < (selstring1[3])[0].length / 2) {

(vp1[2])[5] = !(selstring1[2])[5];

(vp2[2])[5] = (selstring1[2])[5];

} else {

(vp2[2])[5] = !(selstring1[2])[5];

(vp1[2])[5] = (selstring1[2])[5];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane2 = 1;

while (nullbrane2 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp1[3])[2]).push(0);

nullbrane2 = nullbrane2 + 1;

}

nullbrane2 = (selstring1[3])[0].length;

while (nullbrane2 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane2 - 1]);

((vp2[3])[2]).push(0);

nullbrane2 = nullbrane2 - 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1];

(vp2[2])[1] = (((selstring1[3])[0].length - (nullbrane1 - 1)) / (selstring1[3])[0].length) \* (selstring1[2])[1];

vcen1 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane2 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane2 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane2 = 1;

while (nullbrane2 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane2 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane2 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane2 - 1])[2];

nullbrane2 = nullbrane2 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (nullbrane1 < (selstring1[3])[0].length / 2) {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp2[1])[0] = 1 \* Math.PI + seta;

(vp2[1])[1] = Math.PI - sphi;

(vp1[1])[2] = 1 - Math.abs((vp1[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp1[2])[1] == 0) {

(vp2[1])[2] = 0;

} else {

(vp2[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp1[2])[1] / (vp2[2])[1] + 1), 2)));

}

} else {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp1[1])[0] = 1 \* Math.PI + seta;

(vp1[1])[1] = Math.PI - sphi;

(vp2[1])[2] = 1 - Math.abs((vp2[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp2[2])[1] == 0) {

(vp1[1])[2] = 0;

} else {

(vp1[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp2[2])[1] / (vp1[2])[1] + 1), 2)));

}

}

if (Math.abs((vp1[2])[1]) > 1) {

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

} else {

(vp1[2])[2] = 2;

}

if (Math.abs((vp2[2])[1]) > 1) {

(vp2[2])[2] = 2 + Math.floor(Math.abs((vp2[2])[1]));

} else {

(vp2[2])[2] = 2;

}

allparticles.splice(particle1 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

particle1 = particle1 + 1;

}

}

}

nullbrane1 = nullbrane1 + 1;

}

}

}

function attempt\_recombination() {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length && viable == true) {

particle2 = 1;

while (particle2 <= allparticles.length && viable == true) {

selstring2 = allparticles[particle2 - 1];

if (particle1 != particle2) {

nullbrane2 = 1;

while (nullbrane2 <= (selstring2[3])[0].length && viable == true) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring2[3])[0])[nullbrane2 - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring2[3])[0])[nullbrane2 - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring2[3])[0])[nullbrane2 - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring2[3])[0])[nullbrane2 - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring2[3])[0])[nullbrane2 - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring2[3])[0])[nullbrane2 - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring2[3])[0])[nullbrane2 - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring2[3])[0])[nullbrane2 - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring2[3])[0])[nullbrane2 - 1])[8]), 2)];

viable2 = true;

dist1 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist1 < 1e-99) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[0] - (((selstring2[3])[1])[nullbrane2 - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[1] - (((selstring2[3])[1])[nullbrane2 - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[2] - (((selstring2[3])[1])[nullbrane2 - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[3] - (((selstring2[3])[1])[nullbrane2 - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[4] - (((selstring2[3])[1])[nullbrane2 - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[5] - (((selstring2[3])[1])[nullbrane2 - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[6] - (((selstring2[3])[1])[nullbrane2 - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[7] - (((selstring2[3])[1])[nullbrane2 - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[8] - (((selstring2[3])[1])[nullbrane2 - 1])[8]), 2)];

dist2 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist2 <= dist1) {

viable2 = false;

}

if (viable2 == true) {

if ((selstring1[2])[0] == 'open' && (selstring2[2])[0] == 'closed') {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp1[2])[1] = (selstring1[2])[1] + (selstring2[2])[1];

if (Math.abs((selstring1[2])[1]) > Math.abs((selstring2[2])[1])) {

(vp1[2])[5] = (selstring1[2])[5];

} else {

(vp1[2])[5] = (selstring2[2])[5];

}

if (Math.abs((vp1[2])[1]) > 1) {

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

}

(vp1[3])[0] = [];

(vp1[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = 1;

while (nullbrane3 < nullbrane2) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane1;

while (nullbrane3 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

if ((vp1[2])[2] >= 2) {

seta = (selstring2[1])[0];

sphi = (selstring2[1])[1];

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((selstring2[2])[1]) / Math.abs((selstring1[2])[1]) + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[1] + vcen1[1] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[2] + vcen1[2] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

seta = (selstring2[1])[0];

sphi = (selstring2[1])[1];

srho = Math.abs((selstring2[2])[1]) / Math.abs((selstring1[2])[1]);

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[1] + vcen1[1] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[2] + vcen1[2] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = 1;

}

allparticles.splice(particle1 - 1, 1);

if (particle2 < particle1) {

particle1 = particle1 - 1;

}

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

}

} else if ((selstring2[2])[0] == 'open' && (selstring1[2])[0] == 'closed') {

if (unused\_variable == 5) {

vp1 = [selstring2[0], selstring2[1], selstring2[2], selstring2[3]];

(vp1[2])[1] = (selstring2[2])[1] + (selstring1[2])[1];

if (Math.abs((selstring2[2])[1]) > Math.abs((selstring1[2])[1])) {

(vp1[2])[5] = (selstring2[2])[5];

} else {

(vp1[2])[5] = (selstring1[2])[5];

}

if (Math.abs((vp1[2])[1]) > 1) {

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

}

(vp1[3])[0] = [];

(vp1[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane2) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane1;

while (nullbrane3 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

if ((vp1[2])[2] >= 2) {

seta = (selstring1[1])[0];

sphi = (selstring1[1])[1];

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((selstring1[2])[1]) / Math.abs((selstring2[2])[1]) + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.cos((selstring2[1])[0])), (selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.sin((selstring2[1])[0])), (selstring2[1])[2] \* Math.cos((selstring2[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[1] + vcen1[1] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[2] + vcen1[2] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

seta = (selstring1[1])[0];

sphi = (selstring1[1])[1];

srho = Math.abs((selstring1[2])[1]) / Math.abs((selstring2[2])[1]);

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.cos((selstring2[1])[0])), (selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.sin((selstring2[1])[0])), (selstring2[1])[2] \* Math.cos((selstring2[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[1] + vcen1[1] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[2] + vcen1[2] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = 1;

}

allparticles.splice(particle1 - 1, 1);

if (particle2 < particle1) {

particle1 = particle1 - 1;

}

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

}

} else if ((selstring1[2])[0] == 'closed' && (selstring2[2])[0] == 'closed') {

join\_closed\_strings();

} else if ((selstring1[2])[2] == 1 && (selstring2[2])[2] == 1) {

annihilate\_1();

} else if ((selstring1[2])[2] == 2 && (selstring2[2])[2] == 2) {

annihilate\_2();

} else if ((selstring1[2])[2] > 2 && (selstring1[2])[2] == (selstring2[2])[2]) {

annihilate\_3();

}

}

}

if (((viable == true && dist1 < Math.abs((selstring1[2])[1]) + Math.abs((selstring2[2])[1])) && ((selstring1[2])[2] == 1 && (selstring2[2])[2] > 1 || (selstring2[2])[2] == 1 && (selstring1[2])[2] > 1)) && ((selstring1[2])[0] == 'open' && (selstring2[2])[0] == 'open')) {

if ((((selstring2[3])[0])[nullbrane1 - 1] == ((selstring2[3])[0]).slice(-1)[0] || ((selstring2[3])[0])[nullbrane1 - 1] == ((selstring2[3])[0])[0]) && (((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0]).slice(-1)[0] || ((selstring1[3])[0])[nullbrane1 - 1] == ((selstring1[3])[0])[0])) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[0] - (((selstring2[3])[1])[nullbrane2 - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[1] - (((selstring2[3])[1])[nullbrane2 - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[2] - (((selstring2[3])[1])[nullbrane2 - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[3] - (((selstring2[3])[1])[nullbrane2 - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[4] - (((selstring2[3])[1])[nullbrane2 - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[5] - (((selstring2[3])[1])[nullbrane2 - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[6] - (((selstring2[3])[1])[nullbrane2 - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[7] - (((selstring2[3])[1])[nullbrane2 - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[8] - (((selstring2[3])[1])[nullbrane2 - 1])[8]), 2)];

dist2 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist2 <= dist1) {

viable2 = false;

}

if (viable2 == true) {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp1[2])[1] = (selstring1[2])[1] + (selstring2[2])[1];

if (Math.abs((selstring1[2])[1]) > Math.abs((selstring2[2])[1])) {

(vp1[2])[5] = (selstring1[2])[5];

} else {

(vp1[2])[5] = (selstring2[2])[5];

}

if (Math.abs((vp1[2])[1]) > 1) {

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

} else {

(vp1[2])[2] = 2;

}

(vp1[3])[0] = [];

(vp1[3])[2] = [];

if (nullbrane1 == (selstring1[3])[0].length) {

nullbrane3 = (selstring1[3])[0].length;

while (nullbrane3 >= 1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

} else {

nullbrane3 = 1;

while (nullbrane3 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

}

if (nullbrane2 == (selstring2[3])[0].length) {

nullbrane3 = (selstring2[3])[0].length;

while (nullbrane3 >= 1) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

} else {

nullbrane3 = 1;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

}

(vp1[3])[1] = (vp1[3])[0];

if ((selstring1[2])[2] > (selstring2[2])[2]) {

seta = (selstring2[1])[0];

sphi = (selstring2[1])[1];

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((selstring2[2])[1]) / Math.abs((selstring1[2])[1]) + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[1] + vcen1[1] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1])), vcen2[2] + vcen1[2] \* ((selstring2[2])[1] / Math.abs((selstring2[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

seta = (selstring1[1])[0];

sphi = (selstring1[1])[1];

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((selstring1[2])[1]) / Math.abs((selstring2[2])[1]) + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.cos((selstring2[1])[0])), (selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.sin((selstring2[1])[0])), (selstring2[1])[2] \* Math.cos((selstring2[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[1] + vcen1[1] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[2] + vcen1[2] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

}

allparticles.splice(particle1 - 1, 1);

if (particle2 < particle1) {

particle1 = particle1 - 1;

}

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

}

}

}

}

nullbrane2 = nullbrane2 + 1;

}

}

particle2 = particle2 + 1;

}

nullbrane1 = nullbrane1 + 1;

}

if (viable == true) {

attempt\_self\_recombination();

}

}

function join\_closed\_strings() {

if (unused\_variable == 5) {

vp1 = [selstring2[0], selstring2[1], selstring2[2], selstring2[3]];

(vp1[2])[1] = (selstring2[2])[1] + (selstring1[2])[1];

if (Math.abs((selstring2[2])[1]) > Math.abs((selstring1[2])[1])) {

(vp1[2])[5] = (selstring2[2])[5];

} else {

(vp1[2])[5] = (selstring1[2])[5];

}

if (Math.abs((vp1[2])[1]) > 1) {

(vp1[2])[2] = 1 + Math.floor(Math.abs((vp1[2])[1]));

}

(vp1[3])[0] = [];

(vp1[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane2) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane1;

while (nullbrane3 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

if ((vp1[2])[2] >= 2) {

seta = (selstring1[1])[0];

sphi = (selstring1[1])[1];

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((selstring1[2])[1]) / Math.abs((selstring2[2])[1]) + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.cos((selstring2[1])[0])), (selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.sin((selstring2[1])[0])), (selstring2[1])[2] \* Math.cos((selstring2[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[1] + vcen1[1] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[2] + vcen1[2] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

seta = (selstring1[1])[0];

sphi = (selstring1[1])[1];

srho = Math.abs((selstring1[2])[1]) / Math.abs((selstring2[2])[1]);

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.cos((selstring2[1])[0])), (selstring2[1])[2] \* (Math.sin((selstring2[1])[1]) \* Math.sin((selstring2[1])[0])), (selstring2[1])[2] \* Math.cos((selstring2[1])[1])];

vcen3 = [vcen2[0] + vcen1[0] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[1] + vcen1[1] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1])), vcen2[2] + vcen1[2] \* ((selstring1[2])[1] / Math.abs((selstring1[2])[1]))];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = 1;

}

allparticles.splice(particle1 - 1, 1);

if (particle2 < particle1) {

particle1 = particle1 - 1;

}

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

}

}

function annihilate\_1() {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring2[0], selstring2[1], selstring2[2], selstring2[3]];

(vp1[2])[4] = 1;

(vp2[2])[4] = 2;

(vp1[2])[2] = 2;

(vp2[2])[2] = 2;

if (nullbrane1 < nullbrane2) {

(vp1[2])[5] = (selstring2[2])[5];

} else {

(vp1[2])[5] = (selstring1[2])[5];

}

if ((selstring1[3])[0].length - nullbrane1 < (selstring2[3])[0].length - nullbrane2) {

(vp2[2])[5] = (selstring2[2])[5];

} else {

(vp2[2])[5] = (selstring1[2])[5];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2 - 1;

while (nullbrane3 >= 1) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = (selstring1[3])[0].length;

while (nullbrane3 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp2[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1] + ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1];

(vp2[2])[1] = ((selstring1[2])[1] - ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1]) + ((selstring2[2])[1] - ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1]);

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (Math.abs((vp1[2])[1]) < Math.abs((vp2[2])[1])) {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp2[1])[0] = 1 \* Math.PI + seta;

(vp2[1])[1] = Math.PI - sphi;

(vp1[1])[2] = 1 - Math.abs((vp1[2])[1]) / Math.abs(Math.abs((vp1[2])[1]) + Math.abs((vp2[2])[1]));

if ((vp1[2])[1] == 0) {

(vp2[1])[2] = 0;

} else {

(vp2[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((vp1[2])[1]) / Math.abs((vp2[2])[1]) + 1), 2)));

}

} else {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp1[1])[0] = 1 \* Math.PI + seta;

(vp1[1])[1] = Math.PI - sphi;

(vp2[1])[2] = 1 - Math.abs((vp2[2])[1]) / Math.abs(Math.abs((vp1[2])[1]) + Math.abs((vp2[2])[1]));

if ((vp2[2])[1] == 0) {

(vp1[1])[2] = 0;

} else {

(vp1[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((vp2[2])[1]) / Math.abs((vp1[2])[1]) + 1), 2)));

}

}

allparticles.splice(particle1 - 1, 1);

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

if (particle2 > particle1) {

particle1 = particle1 + 1;

}

}

}

function annihilate\_2() {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring2[0], selstring2[1], selstring2[2], selstring2[3]];

(vp1[2])[4] = 1;

(vp2[2])[4] = 2;

(vp1[2])[2] = 1;

(vp2[2])[2] = 1;

if (nullbrane1 < nullbrane2) {

(vp1[2])[5] = (selstring2[2])[5];

} else {

(vp1[2])[5] = (selstring1[2])[5];

}

if ((selstring1[3])[0].length - nullbrane1 < (selstring2[3])[0].length - nullbrane2) {

(vp2[2])[5] = (selstring2[2])[5];

} else {

(vp2[2])[5] = (selstring1[2])[5];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2 - 1;

while (nullbrane3 >= 1) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = (selstring1[3])[0].length;

while (nullbrane3 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp2[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1] + ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1];

(vp2[2])[1] = ((selstring1[2])[1] - ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1]) + ((selstring2[2])[1] - ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1]);

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp1[1])[2] = 1;

(vp2[1])[0] = 1 \* Math.PI + seta;

(vp2[1])[1] = Math.PI - sphi;

(vp2[1])[2] = 1;

allparticles.splice(particle1 - 1, 1);

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

if (particle2 > particle1) {

particle1 = particle1 + 1;

}

}

}

function annihilate\_3() {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring2[0], selstring2[1], selstring2[2], selstring2[3]];

(vp1[2])[4] = 1;

(vp2[2])[4] = 2;

(vp1[2])[2] = 2 + Math.floor(((vp1[2])[1]));

(vp2[2])[2] = 2 + Math.floor(Math.abs((vp2[2])[1]));

if (nullbrane1 < nullbrane2) {

(vp1[2])[5] = (selstring2[2])[5];

} else {

(vp1[2])[5] = (selstring1[2])[5];

}

if ((selstring1[3])[0].length - nullbrane1 < (selstring2[3])[0].length - nullbrane2) {

(vp2[2])[5] = (selstring2[2])[5];

} else {

(vp2[2])[5] = (selstring1[2])[5];

}

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2 - 1;

while (nullbrane3 >= 1) {

((vp1[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = (selstring1[3])[0].length;

while (nullbrane3 >= nullbrane1) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 - 1;

}

nullbrane3 = nullbrane2;

while (nullbrane3 <= (selstring2[3])[0].length) {

((vp2[3])[0]).push(((selstring2[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

(vp1[2])[1] = ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1] + ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1];

(vp2[2])[1] = ((selstring1[2])[1] - ((nullbrane1 - 1) / (selstring1[3])[0].length) \* (selstring1[2])[1]) + ((selstring2[2])[1] - ((nullbrane2 - 1) / (selstring2[3])[0].length) \* (selstring2[2])[1]);

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

(vp2[2])[2] = 2 + Math.floor(Math.abs((vp2[2])[1]));

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (Math.abs((vp1[2])[1]) < Math.abs((vp2[2])[1])) {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp2[1])[0] = 1 \* Math.PI + seta;

(vp2[1])[1] = Math.PI - sphi;

(vp1[1])[2] = 1 - Math.abs((vp1[2])[1]) / Math.abs(Math.abs((vp1[2])[1]) + Math.abs((vp2[2])[1]));

if ((vp1[2])[1] == 0) {

(vp2[1])[2] = 0;

} else {

(vp2[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((vp1[2])[1]) / Math.abs((vp2[2])[1]) + 1), 2)));

}

} else {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp1[1])[0] = 1 \* Math.PI + seta;

(vp1[1])[1] = Math.PI - sphi;

(vp2[1])[2] = 1 - Math.abs((vp2[2])[1]) / Math.abs(Math.abs((vp1[2])[1]) + Math.abs((vp2[2])[1]));

if ((vp2[2])[1] == 0) {

(vp1[1])[2] = 0;

} else {

(vp1[1])[2] = Math.sqrt(Math.abs(1 - Math.pow(1 / (Math.abs((vp2[2])[1]) / Math.abs((vp1[2])[1]) + 1), 2)));

}

}

allparticles.splice(particle1 - 1, 1);

allparticles.splice(particle2 - 1, 1);

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

if (particle2 > particle1) {

particle1 = particle1 + 1;

}

}

}

function attempt\_self\_recombination() {

nullbrane1 = 1;

while (nullbrane1 <= (selstring1[3])[0].length && viable == true) {

nullbrane2 = 1;

while (nullbrane2 <= (selstring1[3])[0].length && viable == true) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[0] - (((selstring1[3])[0])[nullbrane2 - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[1] - (((selstring1[3])[0])[nullbrane2 - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[2] - (((selstring1[3])[0])[nullbrane2 - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[3] - (((selstring1[3])[0])[nullbrane2 - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[4] - (((selstring1[3])[0])[nullbrane2 - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[5] - (((selstring1[3])[0])[nullbrane2 - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[6] - (((selstring1[3])[0])[nullbrane2 - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[7] - (((selstring1[3])[0])[nullbrane2 - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[0])[nullbrane1 - 1])[8] - (((selstring1[3])[0])[nullbrane2 - 1])[8]), 2)];

viable2 = true;

dist1 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist1 < 1e-99) {

dist5 = [Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[0] - (((selstring1[3])[1])[nullbrane2 - 1])[0]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[1] - (((selstring1[3])[1])[nullbrane2 - 1])[1]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[2] - (((selstring1[3])[1])[nullbrane2 - 1])[2]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[3] - (((selstring1[3])[1])[nullbrane2 - 1])[3]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[4] - (((selstring1[3])[1])[nullbrane2 - 1])[4]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[5] - (((selstring1[3])[1])[nullbrane2 - 1])[5]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[6] - (((selstring1[3])[1])[nullbrane2 - 1])[6]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[7] - (((selstring1[3])[1])[nullbrane2 - 1])[7]), 2), Math.pow(Math.abs((((selstring1[3])[1])[nullbrane1 - 1])[8] - (((selstring1[3])[1])[nullbrane2 - 1])[8]), 2)];

dist2 = Math.sqrt(dist5.reduce(function(x, y) {return x + y;}));

if (dist2 <= dist1) {

viable2 = false;

}

if (Math.abs(nullbrane1 - nullbrane2) < 50) {

viable2 = false;

}

if ((selstring1[2])[0] == 'closed' && (nullbrane1 + Math.abs((selstring1[3])[0].length - nullbrane2) <= 50 || nullbrane2 + Math.abs((selstring1[3])[0].length - nullbrane1) <= 50)) {

viable2 = false;

}

if (viable2 == true) {

if (unused\_variable == 5) {

vp1 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

vp2 = [selstring1[0], selstring1[1], selstring1[2], selstring1[3]];

(vp2[2])[0] = 'closed';

(vp1[3])[0] = [];

(vp2[3])[0] = [];

(vp1[3])[2] = [];

(vp2[3])[2] = [];

nullbrane3 = 1;

while (nullbrane3 < nullbrane1) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane2 + 1;

while (nullbrane3 <= (selstring1[3])[0].length) {

((vp1[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp1[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

nullbrane3 = nullbrane1;

while (nullbrane3 <= nullbrane2) {

((vp2[3])[0]).push(((selstring1[3])[0])[nullbrane3 - 1]);

((vp2[3])[2]).push(0);

nullbrane3 = nullbrane3 + 1;

}

(vp1[2])[1] = ((vp1[3])[0].length / (selstring1[3])[0].length) \* (selstring1[2])[1];

(vp2[2])[1] = ((vp2[3])[0].length / (selstring1[3])[0].length) \* (selstring1[2])[1];

if (Math.abs((vp1[2])[1]) > Math.abs((vp2[2])[1])) {

(vp1[2])[5] = (selstring1[2])[5];

(vp2[2])[5] = !(selstring1[2])[5];

} else {

(vp2[2])[5] = (selstring1[2])[5];

(vp1[2])[5] = !(selstring1[2])[5];

}

if ((vp1[2])[0] == 'closed') {

(vp1[2])[2] = 1 + Math.floor(Math.abs((vp1[2])[1]));

} else if ((selstring1[2])[2] == 1) {

(vp1[2])[2] = 1;

} else {

(vp1[2])[2] = 2 + Math.floor(Math.abs((vp1[2])[1]));

}

(vp2[2])[2] = 1 + Math.abs(Math.floor((vp2[2])[1]));

(vp1[3])[1] = (vp1[3])[0];

(vp2[3])[1] = (vp2[3])[0];

if ((vp1[2])[2] > 1 && (vp2[2])[2] > 1) {

set\_direction\_1();

} else if ((vp1[2])[2] > 1 || (vp2[2])[2] > 1) {

set\_direction\_2();

} else {

set\_direction\_3();

}

allparticles.splice(particle1 - 1, 1);

particle1 = particle1 + 1;

viable = false;

allparticles.unshift(vp1);

allparticles.unshift(vp2);

}

}

}

nullbrane2 = nullbrane2 + 1;

}

nullbrane1 = nullbrane1 + 1;

}

}

function set\_direction\_1() {

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (Math.abs((vp1[2])[1]) > Math.abs((vp2[2])[1])) {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp2[1])[2] = 1 - Math.abs((vp2[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp2[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp2[2])[1] / (vp1[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp1[1])[0] = (selstring1[1])[0];

(vp1[1])[1] = (selstring1[1])[1];

(vp1[1])[2] = (selstring1[1])[2];

}

} else {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp1[1])[2] = 1 - Math.abs((vp1[2])[1]) / Math.abs((selstring1[2])[1]);

if ((vp1[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp1[2])[1] / (vp2[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp2[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp2[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp2[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp2[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp2[1])[0] = (selstring1[1])[0];

(vp2[1])[1] = (selstring1[1])[1];

(vp2[1])[2] = (selstring1[1])[2];

}

}

}

function set\_direction\_2() {

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if ((vp1[2])[2] > 1) {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp2[1])[2] = 1;

if ((vp2[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp2[2])[1] / (vp1[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp1[1])[0] = (selstring1[1])[0];

(vp1[1])[1] = (selstring1[1])[1];

(vp1[1])[2] = (selstring1[1])[2];

}

} else {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp1[1])[2] = 1;

if ((vp1[2])[1] != 0) {

srho = Math.sqrt(Math.abs(1 - Math.pow(1 / ((vp1[2])[1] / (vp2[2])[1] + 1), 2)));

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp2[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp2[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp2[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp2[1])[2] = Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2))));

} else {

(vp2[1])[0] = (selstring1[1])[0];

(vp2[1])[1] = (selstring1[1])[1];

(vp2[1])[2] = (selstring1[1])[2];

}

}

}

function set\_direction\_3() {

vcen1 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp1[3])[0].length) {

vcen1[0] = vcen1[0] + (((vp1[3])[0])[nullbrane3 - 1])[0];

vcen1[1] = vcen1[1] + (((vp1[3])[0])[nullbrane3 - 1])[1];

vcen1[2] = vcen1[2] + (((vp1[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen1[0] = vcen1[0] / (vp1[3])[0].length;

vcen1[1] = vcen1[1] / (vp1[3])[0].length;

vcen1[2] = vcen1[2] / (vp1[3])[0].length;

vcen2 = [0, 0, 0];

nullbrane3 = 1;

while (nullbrane3 <= (vp2[3])[0].length) {

vcen2[0] = vcen2[0] + (((vp2[3])[0])[nullbrane3 - 1])[0];

vcen2[1] = vcen2[1] + (((vp2[3])[0])[nullbrane3 - 1])[1];

vcen2[2] = vcen2[2] + (((vp2[3])[0])[nullbrane3 - 1])[2];

nullbrane3 = nullbrane3 + 1;

}

vcen2[0] = vcen2[0] / (vp2[3])[0].length;

vcen2[1] = vcen2[1] / (vp2[3])[0].length;

vcen2[2] = vcen2[2] / (vp2[3])[0].length;

vcen3 = [(vcen1[0] + vcen2[0]) / 2, (vcen1[1] + vcen2[1]) / 2, (vcen1[2] + vcen2[2]) / 2];

if (Math.abs((vp1[2])[1]) > Math.abs((vp2[2])[1])) {

spinrelx = vcen2[0] - vcen3[0];

spinrely = vcen2[1] - vcen3[1];

spinrelz = vcen2[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp2[1])[0] = seta;

(vp2[1])[1] = sphi;

(vp2[1])[2] = 1;

if ((vp2[2])[1] != 0) {

srho = (vp2[2])[1] / (vp1[2])[1];

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp1[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp1[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp1[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp1[1])[2] = 1;

} else {

(vp1[1])[0] = (selstring1[1])[0];

(vp1[1])[1] = (selstring1[1])[1];

(vp1[1])[2] = (selstring1[1])[2];

}

} else {

spinrelx = vcen1[0] - vcen3[0];

spinrely = vcen1[1] - vcen3[1];

spinrelz = vcen1[2] - vcen3[2];

if (Math.abs(spinrelx) < 1e-200) {

spinrelx = 1e-200;

}

if (Math.abs(spinrely) < 1e-200) {

spinrely = 1e-200;

}

if (Math.abs(spinrelz) < 1e-200) {

spinrelz = 1e-200;

}

seta = Math.atan(spinrely / spinrelx);

sphi = Math.acos(spinrelz / Math.sqrt(Math.abs(Math.pow(Math.abs(spinrelx), 2) + (Math.pow(Math.abs(spinrely), 2) + Math.pow(Math.abs(spinrelz), 2)))));

if (spinrely < 0) {

seta = seta + Math.PI \* 2;

}

(vp1[1])[0] = seta;

(vp1[1])[1] = sphi;

(vp1[1])[2] = 1;

if ((vp1[2])[1] != 0) {

srho = (vp1[2])[1] / (vp2[2])[1];

vcen1 = [srho \* (Math.sin(sphi) \* Math.cos(seta)), srho \* (Math.sin(sphi) \* Math.sin(seta)), srho \* Math.cos(sphi)];

vcen2 = [(selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.cos((selstring1[1])[0])), (selstring1[1])[2] \* (Math.sin((selstring1[1])[1]) \* Math.sin((selstring1[1])[0])), (selstring1[1])[2] \* Math.cos((selstring1[1])[1])];

vcen3 = [vcen2[0] - vcen1[0], vcen2[1] - vcen1[1], vcen2[2] - vcen1[2]];

if (Math.abs(vcen3[0]) < 1e-200) {

vcen3[0] = 1e-200;

}

if (Math.abs(vcen3[1]) < 1e-200) {

vcen3[1] = 1e-200;

}

if (Math.abs(vcen3[2]) < 1e-200) {

vcen3[2] = 1e-200;

}

if (vcen3[1] < 0) {

(vp2[1])[0] = Math.PI \* 2 + Math.atan(vcen3[1] / vcen3[0]);

} else {

(vp2[1])[0] = Math.atan(vcen3[1] / vcen3[0]);

}

(vp2[1])[1] = Math.acos(vcen3[2] / Math.sqrt(Math.abs(Math.pow(Math.abs(vcen3[0]), 2) + (Math.pow(Math.abs(vcen3[1]), 2) + Math.pow(Math.abs(vcen3[2]), 2)))));

(vp2[1])[2] = 1;

} else {

(vp2[1])[0] = (selstring1[1])[0];

(vp2[1])[1] = (selstring1[1])[1];

(vp2[1])[2] = (selstring1[1])[2];

}

}

}