|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | | |  |
|  |  | .navigate |  |  | .....................guess what?| .other |this is my first time doing HTML |  |  |
|  | [[home](http://docs.google.com/index.htm)]  [[abstract](http://docs.google.com/abs.htm)]  [[introduction](http://docs.google.com/intro.htm)]  [[hypothesis](http://docs.google.com/hypo.htm)]  [[experiment](http://docs.google.com/exp.htm)]  [[data](http://docs.google.com/data.htm)]  [[conclusion](http://docs.google.com/conc.htm)]  [[we recommend](http://docs.google.com/rec.htm)]  [[daily log](http://docs.google.com/log.htm)]  [[other](http://docs.google.com/other.htm)]  [[bibliography](http://docs.google.com/bib.htm)] |  |  | **Java Graphical Display:**  Using the [[Courseware](http://www.gingerbooth.com/courseware/index.html)] tools created by [[Ginger Booth](mailto:ginger@gingerbooth.com)], we modified the original C++ program and rewrote it in Java, enabling it to be run through the browser as an [[interactive graph](http://docs.google.com/program.htm)].  If you want to see the Java source code or if you have any questions, e-mail [[Chris](mailto:cycrutchfield@home.com)] or [[Brian](mailto:nikeboy1@home.com)].    **Acknowledgments:**  We'd like to thank both Janell Richardson and Steven Richardson for their help in developing the equations and helping us convert the C++ program into Java, respectively.  We'd also like to thank Mr. Thiel for helping us develop our project and developing our hypothesis.  Without the aid of these people, we'd be stuck up the proverbial creek without the proverbial paddle.    **Source Code:**  This is the source code for the first model (in C++):    *#include <iostream.h>*  *#include <fstream.h>*  *#include <apstring.h>*  *#include <stdlib.h>*  *#define CARNIVORE\_BIRTH\_RATE .02*  *#define HERBIVORE\_BIRTH\_RATE .2*  *#define PRODUCER\_BIRTH\_RATE 2*  *#define PRODUCER\_CARRYING\_CAPACITY 100000*  *double min(double a, double b);*  *double max(double a, double b);*  *void main()*  *{*  *double producer, herbivore, carnivore;*  *int time;*  *apstring fileName = "F:\\Documents and Settings\\Administrator\\Desktop\\results\\";*  *apstring temp;*  *cout << "Input initial producer population size (negative for default): ";*  *cin >> producer;*  *cout << "Input initial herbivore population size (negative for default): ";*  *cin >> herbivore;*  *cout << "Input initial carnivore population size (negative for default): ";*  *cin >> carnivore;*  *cout << "Input amount of iterations: ";*  *cin >> time;*  *cout << "Input the file name to be stored to (do not add file extension)" << endl << "(type 'none' to have it output to console)" << endl << fileName;*  *cin >> temp;*  *cin.get();*  *cout << endl << endl;*  *ofstream infile;*  *if(temp != "none")*  *{*  *fileName += temp;*  *fileName += ".txt";*  *infile.open(fileName.c\_str());*  *if(infile.fail())*  *{*  *cerr << "ERROR: Unable to open file " << fileName << endl;*  *abort();*  *}*  *}*  *if(producer < 0)*  *producer = 10000;*  *if(herbivore < 0)*  *herbivore = 1000;*  *if(carnivore < 0)*  *carnivore = 100;*  *if(temp != "none")*  *infile << producer << " " << herbivore << " " << carnivore << endl;*  *for(int t=1;t<=time;t++)*  *{*  *carnivore += (CARNIVORE\_BIRTH\_RATE\*min(herbivore, max(0, 1\*carnivore)/1) - carnivore + min(herbivore, max(0, 1\*carnivore)/1));*  *herbivore += (HERBIVORE\_BIRTH\_RATE\*min(producer, max(0, 1\*herbivore)/1) - herbivore + min(producer, max(0, 1\*herbivore)/1));*  *producer += (((producer - min(producer, max(0, 1\*herbivore)))\*PRODUCER\_BIRTH\_RATE\*(1-producer/PRODUCER\_CARRYING\_CAPACITY)) - min(producer, max(0, 1\*herbivore)/1));*  *if(temp == "none")*  *{*  *cout << "Iteration[" << t << "]" << endl;*  *cout << "Producer population size: " << producer << endl;*  *cout << "Herbivore population size: " << herbivore << endl;*  *cout << "Carnivore population size: " << carnivore << endl;*  *cin.get();*  *cout << endl << endl << endl;*  *}*  *else*  *infile << producer << " " << herbivore << " " << carnivore << endl;*  *}*  *}*  *double min(double a, double b)*  *{*  *if(a < b)*  *return a;*  *return b;*  *}*  *double max(double a, double b)*  *{*  *if(a > b)*  *return a;*  *return b;*  *}*  **This is the source code for the second model:**    *#include <iostream.h>*  *#include <iomanip.h>*  *#include <stdlib.h>*  *#include <time.h>*  *#include <stdio.h>*  *#include <fstream.h>*  *#include <apstring.h>*  *#define CARNIVORE\_BIRTH\_RATE .02*  *#define HERBIVORE\_BIRTH\_RATE .2*  *#define PRODUCER\_BIRTH\_RATE 2*  *#define PRODUCER\_CARRYING\_CAPACITY 10000*  *#define HERBIVORE\_CARRYING\_CAPACITY 1000*  *#define MIGRATION\_RATE\_PRODUCER .1*  *#define MIGRATION\_RATE\_HERBIVORE .05*  *#define MIGRATION\_RATE\_CARNIVORE .01*  *double min(double a, double b);*  *double max(double a, double b);*  *void main()*  *{*  *srand( (unsigned)time( NULL ) );*  *double producer, herbivore, carnivore;*  *int time;*  *double carnivore1, carnivore2, carnivore3, carnivore4;*  *double herbivore1, herbivore2, herbivore3, herbivore4;*  *double producer1, producer2, producer3, producer4;*  *double pprodD1, pprodR1, pprodL2, pprodD2, pprodU3, pprodR3, pprodL4, pprodU4;*  *double pherbD1, pherbR1, pherbL2, pherbD2, pherbU3, pherbR3, pherbL4, pherbU4;*  *double pcarnD1, pcarnR1, pcarnL2, pcarnD2, pcarnU3, pcarnR3, pcarnL4, pcarnU4;*  *double prodD1, prodR1, prodL2, prodD2, prodU3, prodR3, prodL4, prodU4;*  *double herbD1, herbR1, herbL2, herbD2, herbU3, herbR3, herbL4, herbU4;*  *double carnD1, carnR1, carnL2, carnD2, carnU3, carnR3, carnL4, carnU4;*  *apstring fileName = "F:\\Documents and Settings\\Administrator\\Desktop\\results\\";*  *apstring temp;*  *cout << "Input initial producer population size (negative for default): ";*  *cin >> producer;*  *cout << "Input initial herbivore population size (negative for default): ";*  *cin >> herbivore;*  *cout << "Input initial carnivore population size (negative for default): ";*  *cin >> carnivore;*  *cout << "Input amount of iterations: ";*  *cin >> time;*  *cout << "Input the file name to be stored to (do not add file extension)" << endl << "(type  'none' to have it output to console)" << endl << fileName;*  *cin >> temp;*  *cin.get();*  *cout << endl << endl;*  *ofstream infile;*  *if(temp != "none")*  *{*  *fileName += temp;*  *fileName += ".txt";*  *infile.open(fileName.c\_str());*  *if(infile.fail())*  *{*  *cerr << "ERROR: Unable to open file " << fileName << endl;*  *abort();*  *}*  *}*  *if(producer < 0)*  *producer = 10000;*  *if(herbivore < 0)*  *herbivore = 1000;*  *if(carnivore < 0)*  *carnivore = 100;*  *carnivore1 = carnivore2 = carnivore3 = carnivore4 = carnivore/4;*  *herbivore1 = herbivore2 = herbivore3 = herbivore4 = herbivore/4;*  *producer1 = producer2 = producer3 = producer4 = producer/4;*  *if(temp != "none")*  *infile << (producer1 + producer2 + producer3 + producer4) << " " << (herbivore1 + herbivore2 + herbivore3 + herbivore4) << " " << (carnivore1 + carnivore2 + carnivore3 + carnivore4) << endl;*  *for(int t=1;t<=time;t++)*  *{*  *pprodD1 = double(rand())/32767;*  *pprodR1 = double(rand())/32767;*  *pprodL2 = double(rand())/32767;*  *pprodD2 = double(rand())/32767;*  *pprodU3 = double(rand())/32767;*  *pprodR3 = double(rand())/32767;*  *pprodL4 = double(rand())/32767;*  *pprodU4 = double(rand())/32767;*  *pherbD1 = double(rand())/32767;*  *pherbR1 = double(rand())/32767;*  *pherbL2 = double(rand())/32767;*  *pherbD2 = double(rand())/32767;*  *pherbU3 = double(rand())/32767;*  *pherbR3 = double(rand())/32767;*  *pherbL4 = double(rand())/32767;*  *pherbU4 = double(rand())/32767;*  *pcarnD1 = double(rand())/32767;*  *pcarnR1 = double(rand())/32767;*  *pcarnL2 = double(rand())/32767;*  *pcarnD2 = double(rand())/32767;*  *pcarnU3 = double(rand())/32767;*  *pcarnR3 = double(rand())/32767;*  *pcarnL4 = double(rand())/32767;*  *pcarnU4 = double(rand())/32767;*  *prodD1 = MIGRATION\_RATE\_PRODUCER\*(producer1 - min(producer1,max(0,1\*herbivore1)))\*pprodD1/(pprodD1 + pprodR1);*  *prodR1 = MIGRATION\_RATE\_PRODUCER\*(producer1 - min(producer1,max(0,1\*herbivore1)))\*pprodR1/(pprodD1 + pprodR1);*  *prodL2 = MIGRATION\_RATE\_PRODUCER\*(producer2 - min(producer2,max(0,1\*herbivore2)))\*pprodL2/(pprodD2 + pprodL2);*  *prodD2 = MIGRATION\_RATE\_PRODUCER\*(producer2 - min(producer2,max(0,1\*herbivore2)))\*pprodD2/(pprodD2 + pprodL2);*  *prodU3 = MIGRATION\_RATE\_PRODUCER\*(producer3 - min(producer3,max(0,1\*herbivore3)))\*pprodU3/(pprodU3 + pprodR3);*  *prodR3 = MIGRATION\_RATE\_PRODUCER\*(producer3 - min(producer3,max(0,1\*herbivore3)))\*pprodR3/(pprodU3 + pprodR3);*  *prodL4 = MIGRATION\_RATE\_PRODUCER\*(producer4 - min(producer4,max(0,1\*herbivore4)))\*pprodL4/(pprodL4 + pprodU4);*  *prodU4 = MIGRATION\_RATE\_PRODUCER\*(producer4 - min(producer4,max(0,1\*herbivore4)))\*pprodU4/(pprodL4 + pprodU4);*  *herbD1 = MIGRATION\_RATE\_HERBIVORE\*(herbivore1 - min(herbivore1,max(0,1\*carnivore1)))\*pherbD1/(pherbD1 + pherbR1);*  *herbR1 = MIGRATION\_RATE\_HERBIVORE\*(herbivore1 - min(herbivore1,max(0,1\*carnivore1)))\*pherbR1/(pherbD1 + pherbR1);*  *herbL2 = MIGRATION\_RATE\_HERBIVORE\*(herbivore2 - min(herbivore2,max(0,1\*carnivore2)))\*pherbL2/(pherbL2 + pherbD2);*  *herbD2 = MIGRATION\_RATE\_HERBIVORE\*(herbivore2 - min(herbivore2,max(0,1\*carnivore2)))\*pherbD2/(pherbL2 + pherbD2);*  *herbU3 = MIGRATION\_RATE\_HERBIVORE\*(herbivore3 - min(herbivore3,max(0,1\*carnivore3)))\*pherbU3/(pherbU3 + pherbR3);*  *herbR3 = MIGRATION\_RATE\_HERBIVORE\*(herbivore3 - min(herbivore3,max(0,1\*carnivore3)))\*pherbR3/(pherbU3 + pherbR3);*  *herbL4 = MIGRATION\_RATE\_HERBIVORE\*(herbivore4 - min(herbivore4,max(0,1\*carnivore4)))\*pherbL4/(pherbL4 + pherbU4);*  *herbU4 = MIGRATION\_RATE\_HERBIVORE\*(herbivore4 - min(herbivore4,max(0,1\*carnivore4)))\*pherbU4/(pherbL4 + pherbU4);*  *carnD1 = min(min(carnivore1,max(0,1\*carnivore1)), MIGRATION\_RATE\_CARNIVORE\*(carnivore1-min(carnivore1,max(0,1\*carnivore1))))\*pcarnD1/(pcarnD1 + pcarnR1);*  *carnR1 = min(min(carnivore1,max(0,1\*carnivore1)), MIGRATION\_RATE\_CARNIVORE\*(carnivore1-min(carnivore1,max(0,1\*carnivore1))))\*pcarnR1/(pcarnD1 + pcarnR1);*  *carnL2 = min(min(carnivore2,max(0,1\*carnivore2)), MIGRATION\_RATE\_CARNIVORE\*(carnivore2-min(carnivore2,max(0,1\*carnivore2))))\*pcarnL2/(pcarnL2 + pcarnD2);*  *carnD2 = min(min(carnivore2,max(0,1\*carnivore2)), MIGRATION\_RATE\_CARNIVORE\*(carnivore2-min(carnivore2,max(0,1\*carnivore2))))\*pcarnD2/(pcarnL2 + pcarnD2);*  *carnU3 = min(min(carnivore3,max(0,1\*carnivore3)), MIGRATION\_RATE\_CARNIVORE\*(carnivore3-min(carnivore3,max(0,1\*carnivore3))))\*pcarnU3/(pcarnU3 + pcarnR3);*  *carnR3 = min(min(carnivore3,max(0,1\*carnivore3)), MIGRATION\_RATE\_CARNIVORE\*(carnivore3-min(carnivore3,max(0,1\*carnivore3))))\*pcarnR3/(pcarnU3 + pcarnR3);*  *carnL4 = min(min(carnivore4,max(0,1\*carnivore4)), MIGRATION\_RATE\_CARNIVORE\*(carnivore4-min(carnivore4,max(0,1\*carnivore4))))\*pcarnL4/(pcarnL4 + pcarnU4);*  *carnU4 = min(min(carnivore4,max(0,1\*carnivore4)), MIGRATION\_RATE\_CARNIVORE\*(carnivore4-min(carnivore4,max(0,1\*carnivore4))))\*pcarnU4/(pcarnL4 + pcarnU4);*  *carnivore1 += (CARNIVORE\_BIRTH\_RATE\*(min(herbivore1,max(0,1\*carnivore1))/1) + carnL2 + carnU3 - min(min(carnivore1,max(0,1\*carnivore1)), MIGRATION\_RATE\_CARNIVORE\*(carnivore1-min(carnivore1,max(0,1\*carnivore1)))) - carnivore1 + min(herbivore1,max(0,1\*carnivore1))/1);*  *carnivore2 += (CARNIVORE\_BIRTH\_RATE\*(min(herbivore2,max(0,1\*carnivore2))/1) + carnR1 + carnU4 - min(min(carnivore2,max(0,1\*carnivore2)), MIGRATION\_RATE\_CARNIVORE\*(carnivore2-min(carnivore2,max(0,1\*carnivore2)))) - carnivore2 + min(herbivore2,max(0,1\*carnivore2))/1);*  *carnivore3 += (CARNIVORE\_BIRTH\_RATE\*(min(herbivore3,max(0,1\*carnivore3))/1) + carnD1 + carnL4 - min(min(carnivore3,max(0,1\*carnivore3)), MIGRATION\_RATE\_CARNIVORE\*(carnivore3-min(carnivore3,max(0,1\*carnivore3)))) - carnivore3 + min(herbivore3,max(0,1\*carnivore3))/1);*  *carnivore4 += (CARNIVORE\_BIRTH\_RATE\*(min(herbivore4,max(0,1\*carnivore4))/1) + carnD2 + carnR3 - min(min(carnivore4,max(0,1\*carnivore4)), MIGRATION\_RATE\_CARNIVORE\*(carnivore4-min(carnivore4,max(0,1\*carnivore4)))) - carnivore4 + min(herbivore4,max(0,1\*carnivore4))/1);*  *herbivore1 += ((1 - herbivore1/HERBIVORE\_CARRYING\_CAPACITY)\*HERBIVORE\_BIRTH\_RATE\*(min(producer1,max(0,1\*herbivore1))/1) + herbL2 + herbU3 - min(min(herbivore1,max(0,1\*herbivore1)), MIGRATION\_RATE\_HERBIVORE\*(herbivore1-min(herbivore1,max(0,1\*herbivore1)))) - herbivore1 + min(producer1,max(0,1\*herbivore1))/1);*  *herbivore2 += ((1 - herbivore2/HERBIVORE\_CARRYING\_CAPACITY)\*HERBIVORE\_BIRTH\_RATE\*(min(producer2,max(0,1\*herbivore2))/1) + herbR1 + herbU4 - min(min(herbivore2,max(0,1\*herbivore2)), MIGRATION\_RATE\_HERBIVORE\*(herbivore2-min(herbivore2,max(0,1\*herbivore2)))) - herbivore2 + min(producer2,max(0,1\*herbivore2))/1);*  *herbivore3 += ((1 - herbivore3/HERBIVORE\_CARRYING\_CAPACITY)\*HERBIVORE\_BIRTH\_RATE\*(min(producer3,max(0,1\*herbivore3))/1) + herbD1 + herbL4 - min(min(herbivore3,max(0,1\*herbivore3)), MIGRATION\_RATE\_HERBIVORE\*(herbivore3-min(herbivore3,max(0,1\*herbivore3)))) - herbivore3 + min(producer3,max(0,1\*herbivore3))/1);*  *herbivore4 += ((1 - herbivore4/HERBIVORE\_CARRYING\_CAPACITY)\*HERBIVORE\_BIRTH\_RATE\*(min(producer4,max(0,1\*herbivore4))/1) + herbD2 + herbR3 - min(min(herbivore4,max(0,1\*herbivore4)), MIGRATION\_RATE\_HERBIVORE\*(herbivore4-min(herbivore4,max(0,1\*herbivore4)))) - herbivore4 + min(producer4,max(0,1\*herbivore4))/1);*  *producer1 += ((producer1 - min(producer1 ,max(0,1\*herbivore1)))\*PRODUCER\_BIRTH\_RATE\*(1 - producer1/PRODUCER\_CARRYING\_CAPACITY) + prodL2 + prodU3 - min(producer1,max(0,1\*herbivore1)) - MIGRATION\_RATE\_PRODUCER\*(producer1-min(producer1,max(0,1\*herbivore1))));*  *producer2 += ((producer2 - min(producer2 ,max(0,1\*herbivore2)))\*PRODUCER\_BIRTH\_RATE\*(1 - producer2/PRODUCER\_CARRYING\_CAPACITY) + prodR1 + prodU4 - min(producer2,max(0,1\*herbivore2)) - MIGRATION\_RATE\_PRODUCER\*(producer2-min(producer2,max(0,1\*herbivore2))));*  *producer3 += ((producer3 - min(producer3 ,max(0,1\*herbivore3)))\*PRODUCER\_BIRTH\_RATE\*(1 - producer3/PRODUCER\_CARRYING\_CAPACITY) + prodD1 + prodL4 - min(producer3,max(0,1\*herbivore3)) - MIGRATION\_RATE\_PRODUCER\*(producer3-min(producer3,max(0,1\*herbivore3))));*  *producer4 += ((producer4 - min(producer4 ,max(0,1\*herbivore4)))\*PRODUCER\_BIRTH\_RATE\*(1 - producer4/PRODUCER\_CARRYING\_CAPACITY) + prodD2 + prodR3 - min(producer4,max(0,1\*herbivore4)) - MIGRATION\_RATE\_PRODUCER\*(producer4-min(producer4,max(0,1\*herbivore4))));*  *if(temp == "none")*  *{*  *cout << "Iteration[" << t << "] -Sectors-" << endl;*  *cout << setw(12) << "Organism" << setw(6) << " 1" << setw(6) << " 2" << setw(6) << " 3" << setw(6) << " 4" << setw(6) << " TOTAL" << endl;*  *cout << setw(12) << "Producers" << setw(6) << int(producer1) << setw(6) << int(producer2) << setw(6) << int(producer3) << setw(6) << int(producer4) << setw(6) << int(producer1 + producer2 + producer3 + producer4) << endl;*  *cout << setw(12) << "Herbivores" << setw(6) << int(herbivore1) << setw(6) << int(herbivore2) << setw(6) << int(herbivore3) << setw(6) << int(herbivore4) << setw(6) << int(herbivore1 + herbivore2 + herbivore3 + herbivore4) << endl;*  *cout << setw(12) << "Carnivores" << setw(6) << int(carnivore1) << setw(6) << int(carnivore2) << setw(6) << int(carnivore3) << setw(6) << int(carnivore4) << setw(6) << int(carnivore1 + carnivore2 + carnivore3 + carnivore4) << endl;*  *cout << "Press Enter to continue";*  *cin.get();*  *cout << endl << endl << endl;*  *}*  *else*  *infile << (producer1 + producer2 + producer3 + producer4) << " " << (herbivore1 + herbivore2 + herbivore3 + herbivore4) << " " << (carnivore1 + carnivore2 + carnivore3 + carnivore4) << endl;*  *}*  *}*  *double min(double a, double b)*  *{*  *if(a < b)*  *return a;*  *return b;*  *}*  *double max(double a, double b)*  *{*  *if(a > b)*  *return a;*  *return b;*  *}* |  |
|  |  |  |  |  |  |
|  | | | | | |