Answer Sheet Session 3

To be given 3 days before session 3. Students try out the problems first. Then they see this solution and compare it with theirs. We discuss the problems in session 3.

Reminder to TRY IT YOURSELF FIRST

Note that this answer sheet is just a guide to show how things can turn out. The real submission would be the pictures of the results viewed in CURA.

Also note that you should only see this sheet if you CANNOT come up with a code that works at all. Otherwise, your CURA images are enough for a submission.

In fact, the point is to make improvements to the code and make something better than the given code. Put your own spin on it.

This course made to show real work. We start with something relatively simple with minimal instruction and self-learn with online resources. Then we try and keep doing more and more complex things until we have something substantial!

The only difference in real life work is that you don’t get the answer sheet. You make your own!

By the end of this internship, you should add something of your own above and beyond what is mentioned in the answer sheets. That is when you graduate and get ready for the real world!

# Session 3: make a square and rectangle

## Parametric Square

This is just the function to make the parametric square. The rest of the program is almost the same. The function call is also shared later.

Function:

void get\_square(double \*x, double \*y, double \*e, double x0, double y0, double e0, double er, double side)

{

x[3] = x0;

y[3] = y0; // initializing the values

x[0] = x0 + side;

x[1] = x0 + side;

y[1] = y0 + side;

y[2] = y0 + side;

y[0] = y0;

x[2] = x0; // setting x and y values

e[0] = e0 + er\*side;

for(int i=0;i<3;i++)

{

e[i+1] = e[i] + er\*side;

}

Function Call in Main():

// write middle section of code

fileMain.open("sq\_try1.gcode", ios::app);

x0=180;

y0=140;

fileMain<<"\nG1 F1200 X180 Y140 E0 \n"; // setting original position and speed

e0 = 0;

er = 0.013161;

side = 20; // giving initial values for the function

x = new double[4];

y = new double[4];

e = new double[4]; // making the arrays

get\_square(x,y,e,x0,y0,e0,er,side);

for (int i=0; i<4; i++)

{

fileMain<<"\nG1 X"<<x[i]<<" Y"<<y[i]<<" E"<<e[i];

cout<<"\nG1 X"<<x[i]<<" Y"<<y[i]<<" E"<<e[i];

}

cout<<"\nMaking square";

fileMain.close();

//Reading Bottom File

## Parametric Rectangle with Repeated Call

Full program including structure definition and calling the function twice.

#include <iostream>

#include <fstream>

#include <string>

#include <math.h>

using namespace std;

struct coord\_xye

{

double \*x, \*y, \*e; // x n y coordinates and e extrusion values

};

coord\_xye makerec(double x0, double y0, double e0, double l, double b, double er, int num)

{

coord\_xye Crec;

Crec.x = new double[num];

Crec.y = new double[num];

Crec.e = new double[num];

Crec.x[3] = x0;

Crec.y[3] = y0;

Crec.x[0] = x0 + l; // add length

Crec.x[1] = x0 + l;

Crec.y[1] = y0 + b; // add breadth

Crec.y[2] = y0 + b;

Crec.y[0] = y0;

Crec.x[2] = x0; // setting x and y values

Crec.e[0] = e0 + l\*er;

Crec.e[1] = Crec.e[0] + b\*er;

Crec.e[2] = Crec.e[1] + l\*er;

Crec.e[3] = Crec.e[2] + b\*er; // adding up the rectangle

/\* for (int i=0; i<4; i++)

{

cout<<"\nG1 X"<<Crec.x[i]<<" Y"<<Crec.y[i]<<" E"<<Crec.e[i]; // see in function

} \*/

return Crec;

}

int main ()

{

fstream fileRead; // File to read top and bottom

fstream fileMain; // file to write into

string line;

double x0,y0,e0,er,length,breadth; // xy coordinates and extrusion value

coord\_xye coords\_rec; // rectangle coordinates

// Reading Top File

fileRead.open("top\_filler.txt", ios::in);

fileMain.open("sq\_try2.gcode", ios::out); // truncate for the top fill

cout<<"Reading from top filler file\n";

while (getline(fileRead,line))

{

fileMain<<line<<endl;

cout<<line<<endl;

}

fileRead.close();

cout<<"\nTop filler file finished\n";

fileMain.close();

// write middle section of code

fileMain.open("sq\_try2.gcode", ios::app);

e0 = 0;

er = 0.013161;

cout<<"\nEnter x0: ";

cin>>x0;

cout<<"Enter y0: ";

cin>>y0;

cout<<"Enter Length: ";

cin>>length;

cout<<"Enter Breadth: ";

cin>>breadth;

fileMain<<"\nG1 F1200 X"<<x0<<" Y"<<y0<<" E"<<e0<<" \n"; // setting original position and speed

coords\_rec = makerec(x0,y0,e0,length,breadth,er,4); // function call

cout<<"\nMaking First rectangle \n";

for (int i=0; i<4; i++)

{

fileMain<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

cout<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

}

e0 = coords\_rec.e[3]; // new initial value for extrusion

cout<<"\nEnter x0: ";

cin>>x0;

cout<<"Enter y0: ";

cin>>y0;

fileMain<<"\nG1 F1200 X"<<x0<<" Y"<<y0<<" E"<<e0<<" \n"; // setting original position and speed

cout<<"Enter Length: ";

cin>>length;

cout<<"Enter Breadth: ";

cin>>breadth;

coords\_rec = makerec(x0,y0,e0,length,breadth,er,4); // function call the second time with new values

cout<<"\nMaking Second rectangle\n";

for (int i=0; i<4; i++)

{

fileMain<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

cout<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

}

fileMain.close();

//Reading Bottom File

fileRead.open("bottom\_filler.txt", ios::in);

fileMain.open("sq\_try2.gcode", ios::app); // append to add bottom

cout<<"\nReading from bottom filler file\n";

while (getline(fileRead,line))

{

fileMain<<line<<endl;

cout<<line<<endl;

}

fileRead.close();

cout<<"\nBottom filler file finished";

fileMain.close();

}

## Rectangle loop array

Declarations in the main section:

int main ()

{

fstream fileRead; // File to read top and bottom

fstream fileMain; // file to write into

string line;

double x0,y0,e0,er,length,breadth, xinc, yinc; // xy coordinates and extrusion value and x and y increment for array

coord\_xye coords\_rec; // rectangle coordinates

int i,j,num; // number of rectangles

Making user defined loop:

// write middle section of code

fileMain.open("sq\_try2.gcode", ios::app);

e0 = 0;

er = 0.013161;

cout<<"\nEnter x0: ";

cin>>x0;

cout<<"Enter y0: ";

cin>>y0;

cout<<"Enter Length: ";

cin>>length;

cout<<"Enter Breadth: ";

cin>>breadth;

cout<<"Enter x distance: ";

cin>>xinc;

cout<<"Enter y distance: ";

cin>>yinc;

cout<<"Enter number of rectangles: ";

cin>>num;

fileMain<<"\nG1 F1200 X"<<x0<<" Y"<<y0<<" E"<<e0<<" \n"; // setting original position and speed

for(j=0;j<num;j++)

{

cout<<"\nMaking rectangle number: "<<j<<" \n";

coords\_rec = makerec(x0,y0,e0,length,breadth,er,4); // function call

for (i=0; i<4; i++)

{

fileMain<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

cout<<"\nG1 X"<<coords\_rec.x[i]<<" Y"<<coords\_rec.y[i]<<" E"<<coords\_rec.e[i];

}

e0 = coords\_rec.e[3]; // new initial value for extrusion

x0 = x0 + xinc;

y0 = y0 + yinc;

**fileMain<<"\n\nG1 F600 Z2.2 ; lift z axis"; // lifting z axis**

**fileMain<<"\n; new rectangle \nG1 F1200 X"<<x0<<" Y"<<y0<<" \n"; // going to new position with high speed (G0)**

**fileMain<<"\nG1 F600 Z0.2 ; put down z axis"; // put down z axis**

**fileMain<<"\nG1 F1200 ; increase speed again\n"; // increase speed again**

cout<<"\n; new rectangle \nG1 X"<<x0<<" Y"<<y0<<" \n";

}

fileMain.close();

This is again just an example. Think about the flaws in this and see if you can use any smarter ways to parameterize… maybe return some different values in the function? Make a function loop?