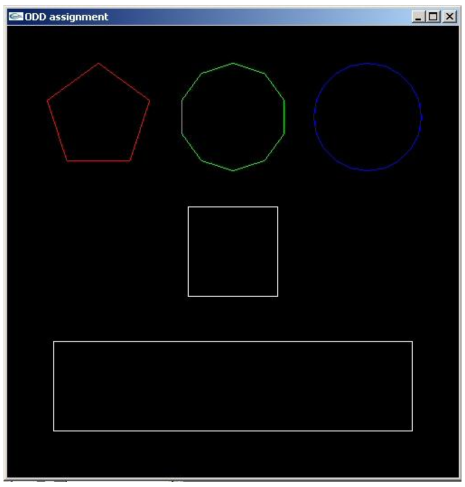
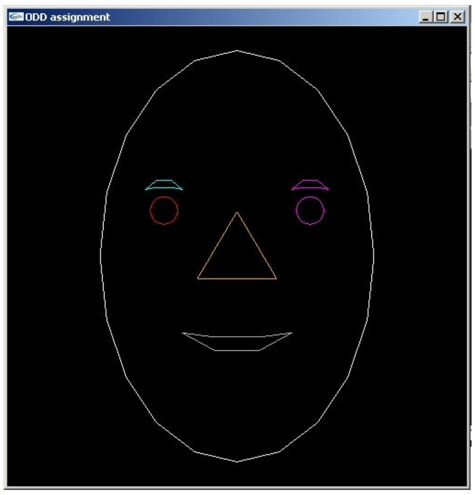
OOD Assignment 2016 – To be performed in groups of three people.

Your task is to write a 2D graphics engine for drawing composite objects out of basic primitives. Your program will read in a file containing a description of a scene in terms of n sided polygons and circles along with 2 basic types of manipulation (a 2D translation and a 2D scale). Your program will then turn this description into a 2D line drawing. Below are two examples for you to test your programs, along with the picture that your program should produce.

File “sample1.txt” can be downloaded from SurreyLearn and should produce the following picture:



File “sample2.txt” can also be downloaded from SurreyLearn and should produce the following picture:



Additional unseen files will be used in testing to see if your program operates correctly. As such you must not hardcode the name of the description file but read it in using command line arguments e.g.

```
        ./myprog sample1.txt
```

You'll be glad to know that you are not expected to build your graphics application from scratch and a sample application has been assembled to simplify the use of graphics and is available along with a sample programme and makefile on SurreyLearn. Please copy it to your local directory and compile it (`make -f makefile_shape`). The program uses openGL and GLUT, although these have been hidden away as much as possible so should not concern you. The objective of this exercise is to learn C++ not openGL. Should you wish to develop upon another platform you will need to install GLUT to use the sample code. GLUT is available for most platforms and in many cases may already be installed.

All the rendering should occur within the draw function (see example), this unfortunately means you may need some global variables but the use of the static keyword will minimise this. You could also use a singleton class. Any initialisation within main should be done before the window object is instantiated.

Some basic openGl commands you will need to know are `glColor3f(r,g,b);` this sets the colour to that specified by r,g,b.

```
glBegin(GL_LINE_LOOP);
        glVertex3f( 0.0f, 1.0f, 0.0f);          // Top
        glVertex3f(-1.0f,-1.0f, 0.0f);          // Bottom Left
        glVertex3f( 1.0f,-1.0f, 0.0f);          // Bottom Right
glEnd();
```

This draws 3 connected vertices. You can have any number of vertices listed between the glBegin and glEnd commands.

`glTranslatef(x,y,z)` translates the cursor on the screen as we are only dealing with 2D you can keep z set to 0.0

`glScalef(x,y,z)` scales the objects being drawn, again as we are only dealing with 2D you can keep z set to 1.0

You shouldn't need to know anything more about OpenGL and rendering than that but should you wish, more details can be found here.

NOTE: your program should open the file once and read it into memory and close the file before entering into the rendering loop i.e. you should not repeatedly read the file each time it is rendered.

Design Specification Hints.

Use polymorphism to build an object which renders differently depending upon the type of object. That object may be a polygon, a circle, a translation or a scaling. Use overloaded input/output operators to read in the content of the file. Use exceptions to catch errors when reading in the file.

Assessment

The project will be assessed in three parts:

1. Initial Design: description of design and specification [20%] Due Week 8 Mon 21/11/16 4pm (16:00).

2. Quality of code; good structure, correct use of language features, clarity of implementation, error checking, good comments [60%] Due Week 11 Mon 12/12/16 4pm (16:00).

3. Working code: demonstration of full implementation [20%] To be performed in laboratory session 13th December.

Submission details:

Title: *Preliminary Design*
Due Date: (Week 8) Mon 21/11/2016 4pm.
Length: *1 Page Max A4 11pt text*
Contents: *The title, names of members of group and login id's. It should then discuss the software design in broad terms: include decisions on the overall software design (with diagram of class hierarchy), the memory management used, error handling and any additional related class and functionality implemented.*

Title: *Final Submission*
Due Date: *(Week 11) Monday 12/12/2016 4pm.*
Length: *1 Page A4 11pt text + compact printout of code*
Content: *A Title page stating team members names and login id's and declaring their contributions to the project i.e. 33% each. Each student should sign this declaration to show they agree with the weight of their contribution. A clear class hierarchy diagram should be included followed by a compact printout of the code.*

Feedback and Marking: The purpose of the initial design is to force the students to think about the problem and solution prior to implementation. Feedback on design will be given to the whole cohort in the lecture (22nd Nov) immediately following submission on the 21st Nov. In this lecture an idealised design will be discussed. This will allow students to judge the suitability of their own designs and make amendments prior to implementation. A mark for the design and further feedback will be provided asap after marking. The final implementation will be marked against the attributes: Design Quality, Code Structure, Use of Language Features, Clarity of Implementation, Error Checking and Comments. Feedback will also be provided on the strengths and weaknesses of the implementation.