University of Southampton	***School of Electronics and Computer Science***	***Coursework (2 of 4) Instructions***
Module: COMP 3004	Title: Principles of Computer Graphics.	Lecturer: Dr. J N Carter
Deadline: 18 November 2011	Feedback: 9 December	Weighting: 10%

Instructions

Introduction

The coursework for this module is in three parts. The first and last are about programming in OpenGL. The first coursework is very restrictive and your aim is to produce images that conform to deliberate specifications. You are required to produce a program which displays things on a number of different screens, under user control

To be successful and obtain 100% (i.e. 10 out of 10) you must meet the specification below. You must submit a single zip file containing the code to build your program.

In this coursework you must produce a number of different screens, selected by typing '`A, B, ..` or `E`' with the following functionality.

A. Draw a wire-frame sphere by calculating all the vertex positions and drawing lines between them.
B. In a similar manner draw a wire-frame cone. It should have a solid base made up of triangles meeting at the centre.
C. For the sphere, work out the surface normal direction[1] and augment your wire-frame drawing with short lines representing the normal direction of each vertex The sphere should now appear to be a hedge hog.
D. Use the normal information calculated in (c) above work out the amount of illumination falling on each vertex. Assume that the light source is at infinity and is co-axial with the viewpoint. Use this to draw a shaded sphere.
E. Develop a simple animation showing a number of cones and spheres moving along regular paths. These can be wireframe or solid.

The program must respond to both upper and lower case. Typing the letter 'Q', 'q' or 'Esc' must cause the program to exit cleanly. The program must start displaying screen A. A successful solution will be demonstrated in lectures.

The following **specific restrictions** apply:

1. Your solution must involve OpenGL Core > 3.1. Mac users may use OpenGL 2.2 if they can not upgrade.
2. Functions in available libraries that draw cones and spheres are explicitly forbidden.
3. Only functions relating to video and keyboard events may be used.

General Restrictions, Rules and Regulations

This exercise is to be marked on a fast PC/Graphics Card combination, using Windows XP and MinGW. However any submissions which are too fast or too slow, i.e. an animation not linked to elapsed time, will lose marks. You must code your animation independent of the frame rate. The code should use the OpenGL and SDL/FreeGLUT/GLWF libraries, and all programming must be done in C or C++.

You have many choices as to what machines to use to do the development on:

- Windows XP/7 using the Microsoft C++ compiler or the GCC compiler.
- Linux using Mesa and gcc

.

- Mac OSX using the GCC compiler.

You must submit a program that will compile and execute on an ECS workstation and compiles without errors or warnings on the hand-in system. If there are irresolvable problems with your work I will ask you to demonstrate it to me in the Computer Lab. Stick to simple solutions, avoid complexity. C++ programmers may assume that the Standard Template Library is available.

The following rules apply:

- **C or C++ only, Objective C, C# or Java are forbidden. It is preferable that you stick to simple C++.**
- **You are recommended to use FreeGLUT or GLWF windowing interfaces. SDL is also an option but it is more complex.**
- **Only .c, .cpp, .h or .hpp and shader files to be submitted. All other types may be deleted automatically.**
- **Don't write C++ and give the file a .C extension. It may not compile properly.**
- **OpenGL or other header files like `gl.h`, `gl3.h`, `sdl.h` or `windows.h` must not be submitted. If you need to do this it is probably because your compute is not set up correctly.**
- **OpenGL Extensions are forbidden.**
- **Any platform specific extensions/features are forbidden[2].**
- **Coursework should run on Windows XP, Linux (Unix) or OS X with recompilation only.[3]**
- **Full Screen mode is forbidden.**

Your program must compile and link on the hand-in machine with out errors or warnings. The compiler will be set so that warnings are treated as errors. If there are any errors then no attempt will be made to run your coursework, and you will receive zero marks.

The on-line checker only checks that programs compile/link.

- If it does so but fails to compile or execute properly on my test machine then you will be given the opportunity to demonstrate a working version.
- If the package as submitted could never have run, e.g. missing, corrupted or incorrect files then a penalty of 20% will be applied to any resubmission, regardless of its correctness or tardiness.
- If your program completely fails to work on the test machine and I am unable to re-build it within a reasonable time, and cannot contact you with in a reasonable time you will be awarded zero for the coursework.

All elements of this coursework must be handed in via the Electronic Coursework Submission system, in a zip file. There are no restrictions on how many times you submit your coursework. However only the last submission before the deadline will be considered.

Normal penalties will be applied for late submission at 10% per day, up to 5 working days. Submission of Coursework 1 after the deadline for Coursework 3 will normally receive zero marks. The usual plagiarism rules apply, and automatic plagiarism detection may be used. However, you are free to borrow from the many examples available, on the course web site or on the WWW, provided you quote your source in your code.

If you use an Integrated Development environment place all the files for this in a suitably labelled directory, but then copy all the code files into the root directory of the zip file. If the

[2] You may not use features of a specific graphics card even if there is an ECS workstation with it fitted.

test script fails to find files where it expects them you will be warned and receive zero for the coursework unless you resubmit with an appropriate file structure.

Required file layout	
No IDE	With an IDE
/	/
+mycode.c	+idecode
+mycode.h	+mycode.c
	+mycode.h
	+ … IDE files
	+mycode.c
	+mycode.h

Names should not be case sensitive, i.e. names like X.c and x.c must not be used together. Names must not clash with system files. Otherwise there are no restrictions on the names or the number of 'c' or 'cpp' files submitted.

Failure to conform to this structure is the most common cause of submissions being rejected.

In some cases you may be asked to resubmit your coursework after the deadline has passed, usually when I judge that there is a problem, there will normally be no penalty in this situation. However re-submission must be by the C-Bass system. I will not accept whole or partial coursework re-submissions by e-mail, disk or other media.

Hints and FAQ (see the web site for more).

- *Note some of these hints may only apply to Coursework 3, especially online.*
- You are highly recommended to submit a test program as soon as the submission system is setup. This will confirm that you understand the procedure and do not have to fix your program at the last minute to ensure it conforms and does not generate errors.
- Keep it simple, don't attempt to build the whole program in one go. Write a number of simple programs to explore all the problems you encounter before attempting to put everything together. In this way you will automatically satisfy the basic requirements.
- Often the biggest problem is getting the viewing right. Make this an early priority.
- Don't be too ambitious, get something to work, submit it, and then build on it and submit again and again.
- Use a structured approach, develop subroutines to draw your primitive objects and then use the geometric transformations to place them in your universe.
- On-line there are plenty of examples, and there are extensive manual pages for all the OpenGL and functions that you might use. Read the relevant manual pages, and then re-read them. There are many more options than those given in the examples and lectures, and if you can discover some of these your task may be easier. Feel free to re-use any of the example code, however failure to attribute your sources will be severely penalised.
- Try and separate modeling from drawing, i.e. don't mix up calculations relating to animation and rendering.
- Any students who feel that their preparation and background will make this project impossible should see me as soon as possible.
- If you have any doubts about how to interpret the specification, either *ask me to clarify the situation* or *make your own design decisions and document them in the source code.*
- The commonest question is '*What aspects of OpenGL can I use in this coursework 1?*'
    - o The answer is simple; as long as it is Core OpenGL you are ok.
- I'm also asked '*Can I use 3rd party (commercial) software to generate my polygon meshes?*'

    - Again the answer is simple, for this coursework, NO. Since part of this coursework is about making your own objects so using a tool to generate them is not appropriate.
- I'm also asked, '*Why cant I get OpenGL to work on my computer?*'?
    - The simple answer is that I don't know as it works on all of mine (Windows XP, 7 or OSX Lion [?]). I'm happy to chat about this after lectures but if you really need help please go to ECS helpdesk.

In general I will answer questions after lectures and will respond to e-mail queries as soon as I can, please mark them "COMP3004 Coursework Query". Any student who wishes to talk to me for a longer period should e-mail for an appointment. If you are not sure about any element of the coursework specification or the rules, talk to me before writing lots of code.

If there are any inconsistencies in these note please let me know as soon as possible.

Submission

Please submit your program in a zip file as specified above, to the C-Bass system. There is no penalty for multiple submissions.

Relevant Learning Outcomes (LOs)

1. Become familiar with the OpenGL and associated libraries.
2. Demonstrate the ability to generate basic shape from first principles.
3. Use a simple interface to control a graphics program.
4. Demonstrate basic concepts of shading, lighting and animation.

Marking Scheme

Criterion	*Description*	*LOs*	*Total*
Function	Demonstrate a functioning graphics program.	1	1
Shapes		2	3
Surface Normal	Demonstrate understanding of geometry and shape	2, 4	1
Shade	Shades sphere showing lighting effects.	4	2
Animation	Combine basic shapes in a simple animation, system	4	2
Control	Switching display and orderly termination.	3	1

Late submissions will be penalised at 10% per working day. No work can be accepted after feedback has been given.
You should expect to spend up to10 hours on this assignment. Please note the University regulations regarding academic integrity.