CMSC 427 Computer Graphics
Programming Assignment 1: Introduction to OpenGL and C++
Due Date: 11:59 PM, February 16, 2017

Project Submission:

1) Delete all intermediate files (run the command make distclean). This will delete the Makefile created during the compilation process. It will also delete the executable file.
2) Place all files for the project in a folder and ZIP up the folder. You will submit your project via the submit server. To submit a zip file, login to the submit server webpage and look for the link to make a *web submission*.

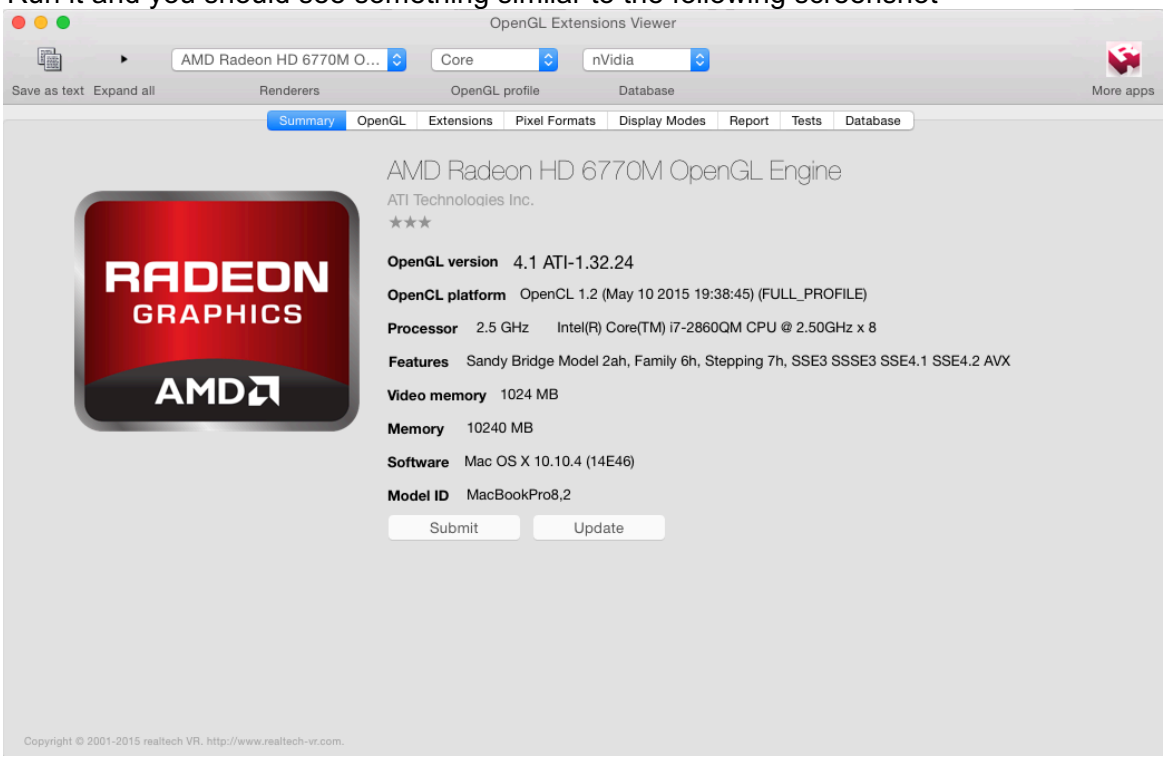
Project Description

The purpose of this assignment is to walk you through the setup/installation process of the various programs you'll need for this course. In addition, you will modify two small OpenGL projects in order to get acquainted with working in both OpenGL and C++. The starter code for these two projects can be downloaded from the submit server. The code in these projects will serve as a template for most of the projects in this course, so it is important that you read and understand what is happening. As always feel free to ask questions in class, in office hours, or on piazza.

Installation and Setup

- The first thing to do is figure out what version of OpenGL is currently running on your computer. This class will require OpenGL version 4.X. Because this is a computer graphics course, we cannot support graphics cards running older versions of OpenGL. If your computer does not have the necessary hardware, you will need to use one of the Mac labs on campus to work on assignments. All coding assignments for this course are required to run on one of the Mac lab computers.

  The quickest way to check what version of OpenGL you have is to download the OpenGL Extension Viewer. This is only available for Mac and Windows.
  Download Link - http://www.realtech-vr.com/glview/
  Run it and you should see something similar to the following screenshot



If you have Linux, you can check your graphics card information from the command line.

```
glxinfo | grep –i "opengl”
```

For Linux users, you may have to look up the specific graphic card on your computer and search online for the corresponding hardware specifications. The vendor documentation should tell you what version of OpenGL is supported by the card.

Note: If your graphics card doesn't support openGL 4.X outright, try updating your device drivers for the card before giving up. In this case you should visit the vendor's website for more information on how to update your drivers.

- For Mac and Linux users, you will need to install gcc, a C++ compiler, if you don't already have it. Be sure to update to the latest version. For Windows users, you will need to install Visual Studio. Visual Studio can be obtained through Microsoft DreamSpark for free with your school email address.

- This class will use a collection of libraries known as Qt. Visit the Qt website (link below) and download the offline installer for your appropriate operating system. Note – the installer will try to get you to sign up for an online account. Skip this step. For this course, we will be using Qt v5.5.0.

  Download Link - https://www.qt.io/download-open-source/#section-3

  This will automatically install an application called QtCreator, an IDE similar to Eclipse. You may choose to work on projects inside QtCreator or from the command line (which is simpler for several reasons).

  We will use the Qt framework quite extensively so you are expected to use the online documentation when you have questions. You should bookmark the documentation now.

  Documentation - http://doc.qt.io/qt-5/classes.html

HelloWorld2D Questions

Download a copy of the starter code called CMSC427_HelloWorld2D. It will contain the following files:

- cmsc427.hpp
- cmsc427.cpp
- cmsc427.pro
- cmsc427.ui
- GLview.hpp
- GLview.cpp
- reachup.png
- resources.qrc
- simple.fsh
- simple.vsh

If you have the correct version of OpenGL and everything installed correctly, you should be able to compile and execute the code. From the command line go to the CMSC427_HelloWorld2D folder and execute the following two commands:

Mac & Linux	Windows
> qmake > make	> qmake > nmake

The qmake program reads the cmsc427.pro file and other configuration files and will automatically generate a Makefile for the project. Running make (nmake) then executes commands found in the Makefile, calling gcc (Visual Studio) and compiling the C++ source code into an executable file called cmsc427. Finally to run your program, execute the command

```
> ./cmsc427
```

Note: the following questions will require you to browse the online documentation.

1) What file contains the main() function, the starting point for the entire application?
2) What classes are inherited by the class CMSC427Win? What classes are inherited by the class GLview?
3) In the function CMSC427Win::keyPressEvent(), certain functions in the glwidget object are called based on what keys are pressed. What is the class type of glwidget? In what class is glwidget initialized?
4) How often is GLview::timerEvent() called while the application is running?
5) How often is GLview::initializeGL() called while the application is running?
6) How often is GLview::paintGL() called while the application is running?
7) In GLview::paintGL(), there is a float variable called g_objScale that is passed to the shader program (called m_shader in the code). We will go into more detail about what a shader program is later in this course. This variable is responsible for defining how far to *stretch* the rectangle along the x axis. Based on the previous questions, how often is the value of g_objScale changed? What equation is used to update g_objScale?
8) The file simple.fsh defines a fragment shader program. A fragment shader is compiled and run on your graphics card independent from the rest of the C++ code running on the CPU. Currently the fragment shader interpolates between the squirrel picture (denoted as texColor1 in the code) and the color spectrum background. Modify the fragment shader code so that the squirrel picture remains visible at all times. (Hint: only one line of code needs to be changed).

2DTriangle Questions

Download a copy of the starter code called CMSC427_minimalGL_2D_twoTriangles. It will contain the following files:

- cmsc427.hpp
- cmsc427.cpp
- cmsc427.pro
- cmsc427.ui
- GLview.hpp
- GLview.cpp
- resources.qrc
- simple.fsh
- simple.vsh

To compile and execute the code from the command line, follow the same directions outlined in the HelloWorld2D example. For some questions, you will be asked to modify code.

9) In the function GLview::initializeGL(), there is an array sqVerts defined as

```
float sqVerts[12] = {
    -.5, -.5, // xy coordinates of the first vertex
    .5,  .5, // xy coordinates of the second vertex
    .5,  -.5, // xy coordinates of the third vertex
};
```

This defines the (x,y) coordinates for each vertex of the triangle. Through trial and error, modify the x coordinate of each vertex until each point of the triangle lies on the left/right border of the application window. What are the new (x,y) coordinates of each vertex of the triangle?

10) In the function GLview::initializeGL(), there is an array sqCol defined as

```
float sqCol[18] =  {
   1, 0, 0, // RGB value of the first vertex
   0, 1, 0, // RGB value of the second vertex
   0, 0, 1, // RGB value of the third vertex
};
```

This defines the color at each vertex of the triangle. The amount of red, green, and blue (RGB) that will be at each vertex is specified in the real-value range of [0,1]. Now modify the code so that

```
float sqCol[18] =  {
   1, 0, 0, // RGB value of the first vertex
   0.8, 0.2, 0.9, // RGB value of the second vertex
   0, 0, 1, // RGB value of the third vertex
};
```

In your own words, explain how this affects the triangle that is drawn. Include a screenshot before the code change and a screenshot after the code change.

11) The file simple.vsh defines a vertex shader program. A vertex shader program is executed and run on your graphics card independent from the rest of the C++ code running on the CPU. In simple.vsh, change the following line of code

```
gl_Position = vec4(aVertex.x * uVertexScale, aVertex.y * uVertexScale, 0, 1);
```

to

```
gl_Position = vec4(aVertex.x, aVertex.y * uVertexScale, 0, 1);
```

In your own words, explain what visual difference this makes to the drawn triangle.

12) While running the application, press the number “2”. Two triangles will be drawn. This is done by the function GLview::moreTriangles(). Modify GLview::moreTriangles() so that four triangles are drawn when the number 2 is pressed. Take a screenshot before the code change (with one triangle) and a screenshot after the code change (with four triangles). The triangles must not overlap each other and they can be any color(s) of your choice except all black (they will not be visible in this case).

13) Modify the code in CMSC427_minimalGL_2D_twoTriangles so that a near-perfect multi-colored circle is drawn using triangles. Hint: near-perfect implies that enough triangles are used to draw the circle such that it *looks* like a perfect circle. Technically, it will still not be a circle. Screenshots of your creation are required for bonus credit.

Code Requirements

Your program must be executable from the command line using qmake and make. For further instructions on how to do this, review the instructions on compiling and executing the HelloWorld2D example.

Grading

Your program will be graded based upon successful calls to it from the command line and a correct implementation. All code that you write should be well commented. Your code is judged subjectively based on the simplicity and clarity of implementation. An implementation that is easy to understand, but has few minor bugs will be scored higher than a messy implementation with the same number of minor bugs.

Documentation Grading

Please include documentation in the form of a Markdown (https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet) with your assignment. You will additionally be graded on the completeness, clarity, and conciseness of your documentation. The document, where needed, should included figures and links providing additional details and references.