CS334: Assignment #0 – Cook it!
OpenGL/FreeGLUT Warm Up

Out: Jan 26, 2021
Back (i.e., due): Feb 2, 2021

In order to learn the essential functions of OpenGL without dealing with low-level configuration, we will use FreeGLUT library which is a free-software/open-source alternative to the OpenGL Utility Toolkit (GLUT). FreeGLUT code can be compiled for generating the dynamic library for Windows, Linux and Mac.

NOTE: if you *really* want to use Qt, that is an option. Please contact instructor ASAP.

In [http://www.transmissionzero.co.uk/software/freeglut-devel/](http://www.transmissionzero.co.uk/software/freeglut-devel/) you can download an already compiled version of the library for Windows machines. A suggestion: if you really desire to work on a non-Windows machine, you could do all your development on your PC and then come to B160 to do whatever minor tweaks are needed to get the program to compile and run. However, it is *your* responsibility to ensure the program runs on LWSN B160 machines – if it does not, your program is considered to ‘not run’ which has severe score penalties.

Objective:
The objective of this assignment is to create a simple warm-up program using OpenGL and FreeGLUT programming and graphics environment. This assignment will require you to minorly edit and then compile a provided OpenGL/FreeGLUT application that mostly makes direct calls to OpenGL. It is to your benefit to organize your programming environment as you see fit. You have one week but it should take you much less time.

Important:
Run the provided framework in the PCs of the lab or in your own PC and check that it runs without problem. You can add your own .h and .cpp files to the project, but keep in mind not to modify the basic structure of the project (location of the GL folder and the freeglut.dll and freeglut.obj files) otherwise the integrity of the project will be compromised and may not run properly. The logic of the code can be grouped in three categories: 1) Rendering using OpenGL, 2) Events and 3) A simple menu for UI. Check the code and the comments for additional details. More information concerning OpenGL and FreeGLUT will be given in the PSO sessions next week.

The assignment is to implement a program which draws a simple 2D screen saver like program. The “screen-saver” consists of two squares bouncing within the confines of the window (i.e., within a rectangle) and one line segment being drawn between the squares. To draw the line, simply define an array for the two points and draw it using the glDrawArrays function (see the example on the framework). The endpoints of the line should be within the squares at all times; it can be at a relatively fixed position within each square or moving around – up to you! The velocity of the squares may be increased
or decreased by the user by using a menu-like UI (again, see the example on the framework as a starting point). All the elements should bounce internally in the window and move continuously at a reasonable speed.

Specifics (note: each item is labeled with a unique number and the percentage of the grade it accounts for)

1. (34%) Run the provided framework and add a second bouncing square with its own variables and behavior. Both squares must start at random locations within the window and with random (but reasonably) initial velocity vectors.
2. (33%) Add the line that is always between the two squares. Remember that for any location of the squares this line is always right between them. The line should have a reasonably length.
3. (33%) Implement a menu-like UI for increasing and decreasing the velocity of the squares.

Code Template:
For this assignment, the code framework is on the course website in BrightSpace.

Handing-in Assignments:
To submit the assignment, please use Purdue’s BrightSpace tool. Your complete Visual Studio project (e.g., project file, data files, source code and precompiled executable) is due before class time on the due date. It is your responsibility to make sure the assignment is delivered on time. **Hint: don’t wait until the last moment to hand in the assignment!**

For grading, the program will be run with no command line parameters and tested against the above specifics. When appropriate, the code will be looked as well for proper functionality. If the program does not compile, *zero* points will be given.

Copying/Plagiarism:
As a reminder, **ALL assignments (except the final assignment and upon instructor approval) are individual assignments. This means YOU MUST DO YOU OWN PROGRAMMING. If a copy case is found, both the provider and copier of the assignment will be given a grade of “0” for the assignment and the case will be reported to the University level office which will evaluate the case – expulsion from Purdue is a very likely outcome. Thus, please do your own work. You may consult the TAs, and your peers, for technical questions but not for copying solutions.**

If you have more questions, please see myself or the TA.

Good luck!