**Name**: Haczar Criollo

**Title**: Assignment 1: OpenGL Initialization

**ID**: 2895038

**Date**: 2/28/19

**Description of Program:**

This program was designed to be an OpenGL environment that can load obj formatted models, assign material, and gets affected by a light source. You can also apply transformations like rotations, translation, scaling.

**Readme to illustrate how to run the program**:

Controls:

* *Escape*: Quit the program and then the console will require you to press any key in order to exit completely.
* *W,A,S,D*: Forward, Left, Backward, Right Camera Movement (respectively)
* *Left Shift / Space*: Downward, Upward Camera Movement
* *R*: **Hold** to rotate model
* Arrow Keys (↑, ←, ↓,→ ) : Translate model up, left, down, right (respectively)
* - /+: Scale model

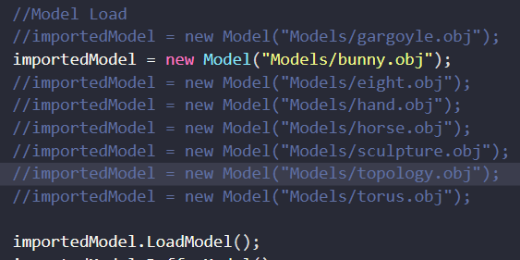
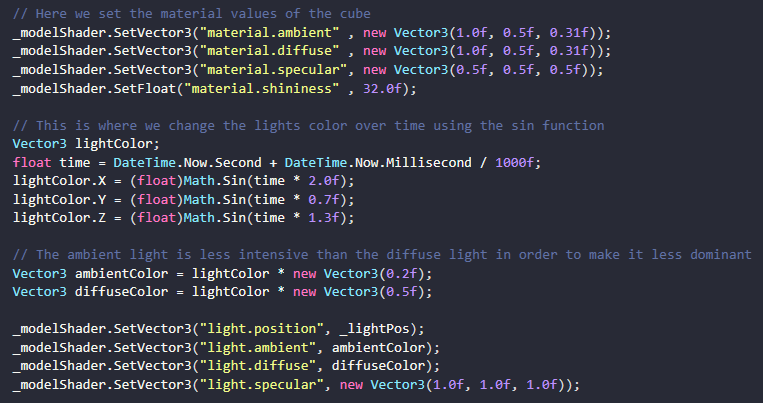
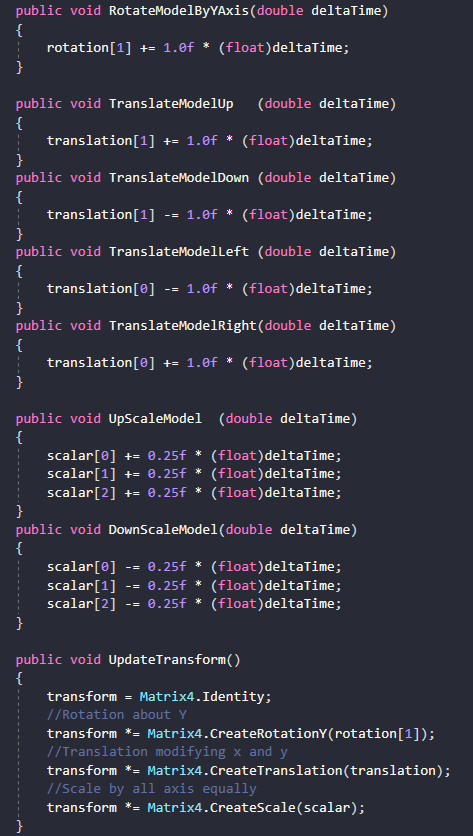
The program will be in this zip file that you download from canvas: 

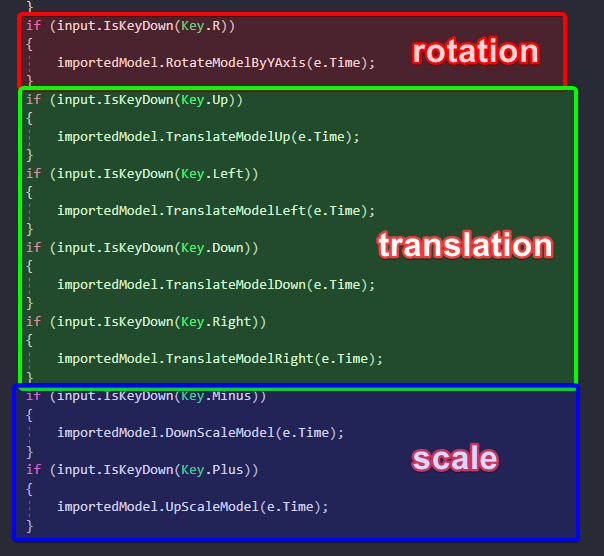
First extract the zip to a folder and go that folder and navigate to   
OpenGLCompGraph\bin\Debug\ OpenGLCompGraph.exe

If you want to change models you will have to go into the code:

1. Open \OpenGLComphGraph.sln using Visual Studio 2019 (the version that I used)
2. Go to Solution Explorer and open Window.cs
3. Line 47 – 54 is where the model is chosen from the objs inside of the models folder
4. Comment the currently uncommented line to hide the bunny.obj and uncomment a new model that you wish to view.
5. Save Window.cs and run the project from VS2019 or build and save the project and open the executable at the filepath written above

**One worked case for each operation**:

* The environment was made using OpenTK which contains fast, low-level bindings for OpenGL using C# which I am comfortable with. It provides several utility libraries, including a math/linear algebra package, a windowing system, and input handling.
* In the environment each 3d model is loaded at default scale, transform, and rotation.
* OnLoad we will get the obj file and use the LoadModel() function
  + This reads the file and populates a vertex and face array and attributes x,y,z to each corresponding vertex and face
* Materials and lighting are set in OnRenderFrame()
  + 
  + Rotation, translation, and scaling are all done manually by the user of the program so they are set to key binds in OnUpdateFrame() which calls a function in our model class that updates the model space accordingly



Summary:

* Done
  + I believe the assignment is complete. I can use the environment I have now to expand on functionality and I believe this is a great starting point.
* To-Do things
  + I would like to implement rotating on all axes and better binds. I like to bind to numpad but I did not for this project just in case the user does not have a numpad on their keyboard.
* What I learned
  + I learned to install packages from nuGet using visual studio, add dependencies, keybinding, and I feel more comfortable with OpenGL than I ever have before.
* The problems met along the way.
  + It was pretty difficult to parse the obj file in order to apply the values to vertex and face array but once it was done it was very easy to render.
  + It was also slighty difficult to update the model transform because I was not connecting the uniform model variable from the shader to my imported model. Once I was able to access the parameter from my shader, I was able to manipulate it easily.
* Remaining Questions
  + I would like a breakdown of all the parameters for different file formats like obj, fbx, collada, stl, and the newly created glTF. With that, I believe it will be easier to understand why there are so many different file formats for 3d models.