Jonathan Boeglin

CS-330 Week 7

Project Reflection

**Design Decisions**

My scene consists of a desk, a drawer, the legs of the desk, a light on the desk, the floor, and two accent lights. These objects cover the necessary primitive shapes by representing a torus, cube, plane, and pyramid. I chose to model the desk as two torii linked together, with the drawer inside as a separate object. The drawer was a cube, as the inside did not need to be modeled. The legs of the desk were also two cubes, and the desk light was a pyramid. The floor and the two accent lights were both planes, as they did not need to convey any sort of depth and were just needed to display the rest of the scene. The way I programed this scene was by splitting the lights and the other objects into two different shader programs, and then by using unique functions to build each object. The drawer, desk, legs, lights, desk light, and floor all had unique functions as the vertices needed to create each were different.

To start, the user can exit the scene at any time using the escape key to close the window. For navigation, the user can move forward, left, backward, and right with the w, a, s, and d keys, in that order. The user can also change heights with the q and e keys. In addition, the perspective can be changed to and from orthographic by pressing the v and b keys. The camera can be turned based on mouse movement, so that the scene can be viewed from different angles and sides. Finally, the scroll wheel allows the speed at which the camera moves to increase and decrease. This was set up using the camera header and object, and by mapping the correct keys to the desired inputs in the input detection functions. One issue that I had while mapping these keys is that originally the switch for orthographic was handled by a single key, but the presses were being registered continuously instead of only when initially pressed, making it hard to land on the different perspectives. This was solved by splitting the operation between two keys and making the perspective shown based on a Boolean.

Some of the custom functions used include the shader program, the objects, and the texture application. I was able to create multiple different shader programs, one for the objects and one for the lights. Doing this in-line would have meant re-creating the shader code for each object as well. I was also able to easily load different textures to the program and apply them to the shapes I wanted. Keeping this texture code separate helped make it easier to understand what textures were being applied where. Finally, creation of multiple objects of the same kind, like the two accent lights, was made easier by not having to re-type the creation code. If I ever need to create lights or other primitive shapes in OpenGL in the future, I will now have a basis to work from.