Matthew Trembley

6/19/2022

Southern New Hampshire University

CS-330

Final Project Reflection

The final project for CS-330 was a culmination of skills learned in past courses, coupled with what was taught in this course alone. There were many development choices that played a factor in the outcome of my project, but the decision I made was to create objects on a scale of easy to moderately difficult. In my 3D scene, I needed to render a printer, a mug, and a faux plant. Before any knowledge of openGL, I felt that it was going to be simple, and I would be given all the tools needed to successfully do it just as I pictured. Each object had its own difficulties, but between each week I was able to finally overcome them. I was easily able to program the required functionality by closely following the tutorials given by SNHU. This helped me be able to focus on the creation and imagery of my 3D scene rather than getting the functionality to work because the tutorials saved so much time.

A user can easily navigate the 3D scene by using common controls of W for forward, S for backward, A for left, and D for right. Q will allow the user to move downward, and E will allow the user to go upward. The mouse allows the user to look around, using the scroll wheel to change the speed in which the camera moves. The set up for control of the camera was only set up for using a mouse and keyboard, as the program itself is still somewhat primitive compared to today’s standards. With a lot more time and devotion, possibly a touchscreen or other input types could be accepted into navigating the world.

Text

Description automatically generatedEncapsulation is one of the fundamentals of object-oriented programming and it is the practice of bundling data with methods that operate on said data. I did my best to write modularized code that could be used in other projects. For example, SNHU wrote a section of code for loading in textures:

This piece of code captures my idea of modularized code, as it creates loading any texture a LOT cleaner with a function call of loadTexture(“filepath”). This is just one example of many in my source code in which I use a function to create clean, readable code that will allow for any other user to come in and understand the functionality of the code.

A function I personally developed was: Text

Description automatically generated

Although not completely on my own, the development of this function enabled for it to be called at the beginning of the main function with a call of UProcessInput(Window). This allows for much cleaner code, especially inside the main function which already houses the “Render” loop which is extensive as it is. The two functions above are technically able to be copy and pasted into another project, allowing for another project to be more readable.