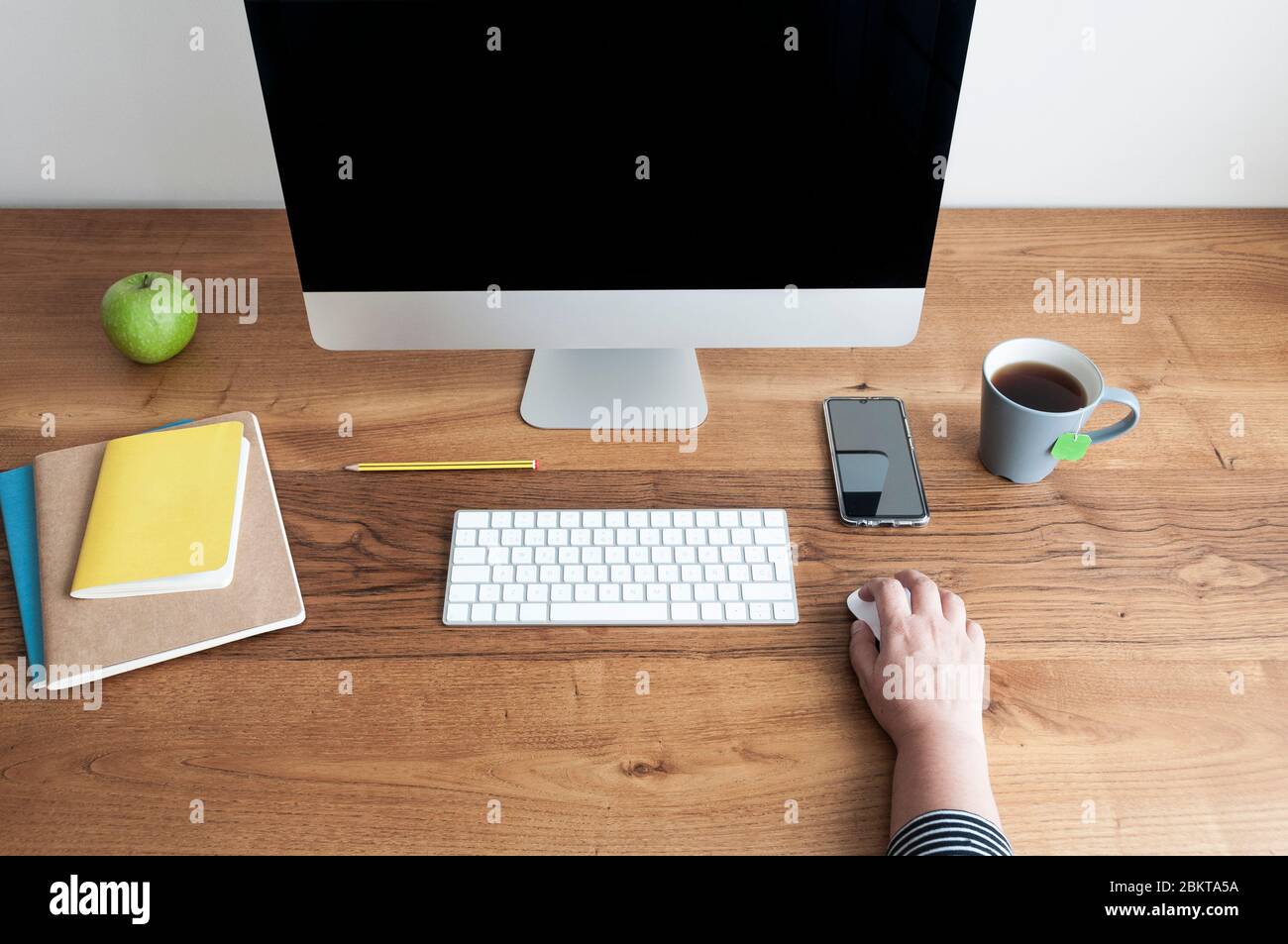
CS 330

Mathew Denison

8/14/2022

Final Project Reflection



The set of items that I have chosen include are a keyboard, pencil, screen and a phone. There are other items, but these are the ones that I will be modeling.

Given the requirements of the program, I was able to model the objects in a 3D space. I was able to create the keyboard with cubic shapes that had their own textures to match the colors of both the keys and the keyboard body. The most difficult part was finding a texture that worked with the keyboard and apply the texture to it. Instead of creating a single cubic object, I decided to add vertices for each key along with the keyboard body. This gave me flexibility in how I applied the textures so that it met the color requirements of the object. It was not how I originally envisioned the keyboard to be displayed but it seemed to have worked out alright.

The controls/navigation of the 3D world were actually the most simple for this assignment. What I did was decide to have the actual changing of the camera movement done within the camera.h file. I created methods in this file that would move the camera in a certain direction (x and y axes) and then called those methods with a keyboard input function in the source file. This allowed for simplicity of the code along with it being more readable within the source file, as all camera movement functionality was through the header file. Another requirement that was needed for the movement is to take input from the mouse which adjusts the viewpoint of the camera. To do this I simply called the MouseButtonCallback function that is recognized through the source code for openGl. I then called this method and placed the x, and offsets to be changed by the mouse. I also decided that it was necessary to have a first mouse option where it places the mouse at a specific location. This keeps the view of the objects in site when first starting the program. The mouse then receives movement functions and the camera changes by calling ProcessMouseMovement.

The functions that I created to allow the project to be more readable and ease of development was the use of pragma regions. This allows me to control the shaders and when they are run. This enabled me to create individual objects in the scene with their own vertex arrays. It also made the code more modular as each time that I wanted to add in a new object with a shader, I simply needed to reference where the pragma regions were created. I can then copy and paste the code from the pragma regions, updating which vertex array to draw, and the program does exactly that. It did end up creating more lines of code than could have been built with a consolidated array, but it gave me more control later on shaders that were called, texture data, and colors.

Overall, while I think that I learned a ton about openGl this semester, I also think that I have quite a bit to learn yet. There is so many different ways to achieve the desired outcome including options like GLUT which seems to make the development process easier. I am excited to write more code in the future with OpenGL as I create more projects and possibly dive into game development. I hope that my understanding from this class lends some future insight into easier/simpler methods to achieve the desired outcome of a 3D scene.