Justification:

The 3D scene I chose at the beginning of the semester was a raspberry pi 4, a rubiks cube, a lip balm, and a Nintendo switch screen. At the time, I was rather new to the OpenGL functionality. I chose these objects because they resemble primitive objects of rectangular prisms, cubes, and cylinders. Unfortunately, I had no idea on the difficulty of making a cylinder 3D object using OpenGL. I understand that the logic in creating a cylinder is to create a circular plane of several sectors. It was difficult for me to replicate into code. I decided to create the lip balm using a rectangular prism. The following list shows how each object is created:

Nintendo Switch screen:

It is composed of two rectangular prisms. The first prism doesn’t have the screen. The second prism contains the screen. Each prism overlaps each other in the 3D space.

Rubiks cube:

I chose structures to define the rubiks cube. I decided to separate the rubiks cube by 6 meshes or 6 faces the rubiks cube. This is so that each texture of the rubiks cube represents the different faces of it.

Raspberry Pi:

I also chose structures to define the Raspberry Pi. I grouped the Raspberry Pi into 3 segments: The top part, the bottom part and the body. The top part contains the texture of the logo. The bottom texture contains the vents of the Raspberry Pi case.

Lip balm:

I also chose structures to define the lip balm. I was intended to use a cylinder as the lip balm, but soon realize it was difficult for me to implement in code. I decided to create a rectangular prism. The body contains the wrapper of the lip balm. The top is the cap. The bottom is just white.

Plane:

The plane is a wooden table. I chose a rectangular prism with a height of 0 to define the plane.

Directional lights:

There are two directional lights in the 3D scene. One Light on the top. The other light on the bottom. This is to provide maximum lighting of the scene. Structures were also used to define the lights. Each light contains normal white lighting.

User Navigation:

WASD is used to pan the camera left, right, up, and down. The mouse pointer is used to point the camera view of the 3D scene. Q and E is used to make the camera go up and down in the y-axis. The mouse scroll wheel is used to increase or decrease speed of movement. By using the camera.h from the LearnOpenGL library, I was able to modify the user-interface of the camera. The program would listen to keystrokes and mouse movements. Then it would determine what to do with the user-input.

Custom Functions:

As I got better at coding in OpenGL, I found that it was easier to define structures to classify the different 3D objects in the scene. I used structures to define directional lighting, the rubiks cube with various faces, and the raspberry pi. Defining structures or even incorporating object orientation later would help make the code more readable and reusable in the future.