**Final Project: Triangle and Cube Studios**

**SNHU: CS-330**

**Nathan Book**

**10/20/24**

**Reflection:**

My approach to selecting pre-defined shapes for the 3D construction of my 2D scene was a simple question that I repeated to myself. What shapes could be combined to form the objects in my scene? For the tree stump, my choice was simple, a cylinder or tapered cylinder. Trees do not have a set shape, but either of these two shapes will come close to its real-world resemblance. For simplicities sake, I chose the cylinder because I did not want to redefine the vertices for the top part of the tapered cylinder to more closely resemble a stump rather than a tipless cone. For the axe, I chose a prism as the blade, a box as the head, and a cylinder as the handle. I combined the prism and box to make an object that resembled the head of an axe stuck in the stump. The objects share the same texture and lighting to help create the illusion of one object. While the handle is a scaled cylinder that fits inside the box object (axe head) with no overlap or clipping. For the wood log I used a cylinder with a bark texture. For a more realistic looking wood log, I would need to learn how to apply two different textures to the same object based on vertices. For this object, I would need the wood grain texture on the top and bottom of the cylinder and the bark texture wrapped around the sides. I was unable to figure this out so the entire object is just bark textured. For the last object, the split wood, I used a rough wood texture that resembles the inside of a tree, which would fit the split wood profile. The positioning, sizes, textures, shading, and lighting were all configured based on OpenGL functionality. These configurations were input at an extreme high or low then slightly adjusted several times over to achieve the required integers for those specifications. Checking the output of the float and double types used to manipulate a digital scene was where the bulk of my time was spent during these assignments. This allowed me to learn the effects of each of these functions on the objects in a scene.

To navigate the scene is relatively simple. Once loaded in, you will be able to move the camera forward, backward, left, right, up, and down using the WASD and QE keys for movement. To make this possible, I had to write code that assigned each of these keys with an action and output. The mouse scroll wheel is used to slow down or speed up the movement of the camera within the scene. For the camera to respond to input functions, each input was assigned a callback function (keyboard, mouse, etc.). This callback function is constantly updating the camera vertices during input.