My 3D Scene

The scene that I selected consisted of a square case, a keyboard, a pyramid, an ink bottle, a mouse, and a few other things in the background. I have a desk light that I typically use for extra light when I have something to work on via my laptop. When I started this course, I could see how the lighting and shading affected the objects that I had resting on my desk. Seeing how a shadow was formed behind some of the objects that I had placed on my desk, made me more curious when I realized that In some of our assignment examples, we had the exact shadow effect within that particular scene. I knew at that moment, that I could at least attempt to capture the scene on my desk, and bring it to life on my pc. This is the reason that I selected the scene that I did.

The way that I got started was through the scene, as far as loading the different shapes, was with the SceneManager::PrepareScene(). Within this scene, you have to load the meshes. The meshes represent the shapes that you want to load into your program. When I say load, I am referring to the shapes that will display in your scene when you run it.

Next, you will go down to “RenderScene”. Here is where you will initiate the rendering of your shape. Y can identify the mesh that you want to render, or draw. You also assign the type of material that you want your material to appear to be made out of. You also specify things like rotation degrees in each of the three axis as far as the X, Y, and Z axis. Once you have identified the shapes that you want to render or display in your scene, you will insert “m\_basicMeshes->DrawPyramid4();” to identify the shape that you want to render. The “scaleXYZ = glm::” will allow you to control the size of your pyramid in X, Y, and Z. With this function you can make the pyramid longer, wider, or taller.

Let’s talk about navigation. The controls that allow a user to navigate through my scene is set up regarding the WASP button methodology. The “W” moves forward, “A” moves to the left side, “S” moves you back from the scene, and “D” moves you to the right. The “Q” button will move you up, and the “R” button will move you down. This set up here along with the scroll function will allow you to pan in any direction that you so choose within the scene.

The code that I developed, is broken down as it should be. It is really straight forward. The different material assignments, lighting, and frag mentors are identified with comments all along the code. Even the code that identifies each individual shape is identified. There should not be any issue for a developer to figure out what was done/