Considering how new all these exercises were to me, while choosing my objects for my scene I wanted to find items that were visually interesting, while not being too complicated to build out. My scene contains several small objects of varying complexity, as I wanted to work on easier objects at first and build up to the more complicated objects that were made up of multiple simple shapes. Take the dice object for example. It’s the simplest of the scene, as it’s just a cube primitive and nothing else. I used this object to learn how transparency works for the shader color call back. The glue jar object was similar in principle to the die, but it added more complexity since it was made up of multiple primitive shapes. The paint pot object was the first of the multi-primitive objects that I attempted to place textures on. Unfortunately, I couldn’t get my personal photos to import into the program. I think the files might have been too large or formatted incorrectly. I used a blue texture found in the utilities folder, but I’m not entirely happy with the result. That last object is the book. This is the most complicated object; in that it’s made up of the greatest number of primitives to get the shapes right. It’s also two different textures and materials working together. I’m happy how the leather texture and material worked out. As well as the texture and materials for the paper. The orientation looks right, and the result was a decent looking simulation of a leather-bound book.

My project currently only supports mouse and keyboard inputs. First, the display window needs to be created. Then there are functions for tracking the position of the mouse. The program then calculates a delta time between rendered frames at which the mouse movement from one frame to the next is calculated and the camera is adjusted accordingly. This delta time is used for keyboard inputs as well. At every frame, the process keyboard events function checks if an input command was sent. The camera movement is calculated by the direction that the key was assigned by the amount calculated by the delta time.

While building out the camera functionality I needed to create a function to handle inputs from the mouse wheel. This function is called every time the mouse wheel input command event happens, and either increases or decreases the delta time calculation depending on direction. Thus, speeding up or slowing down camera movement speed. The ProcessKeyboardEvents functions works by checking if any input commands have been sent, and depending on the input key the camera is moved by that direction calculated by the delta time from the last frame. The render scene function gets passed in all the mesh, texture and materials created, and processes their vector locations to draw the scene every frame.