7-1 Final Project

The scenario I chose for this final project is my dinner table including three objects: my old laptop, a trophy I received as Associate of the Month at my current job, and a creatine bottle. Each of these objects holds special meaning for me. The old laptop symbolizes my perseverance in completing assignments despite having a terribly slow and non-portable laptop with a dead battery. The trophy represents my hard work and recognition in my job. Finally, the creatine bottle signifies my commitment to maintaining a healthy lifestyle and staying active. In terms of programming these objects, I started by learning the basics of OpenGL, which included how to render objects, apply textures, and create lighting effects. This knowledge allowed me to create and display each of the items in my 3D scene. I focused on finding and editing detailed textures for the laptop, trophy, and creatine bottle to enhance their realism. Additionally, I put lighting to illuminate the objects, which adds more realism to the scene. Hence, these objects were represented in the program as closely as possible to their real-life counterparts by applying detailed textures and reflections.

The program allows users to navigate freely through the scenario using a keyboard and a mouse. I achieved this by implementing a camera that responds to user inputs. The camera can move forward, backward, left and right using the WASD keys, and up and down using the QE keys. Additionally, the mouse controls the camera's pitch and yaw, allowing the user to look around the scene in any direction. To set up the camera, I used OpenGL's camera functions to track the camera's position and orientation in the 3D space. The keyboard inputs were mapped to update the camera's movement values, adjusting its position. Also, mouse inputs are recognized to modify the camera's rotation. With all these, users can explore the scene from various perspectives.

In my program, I implemented a feature that allows the user to switch between orthographic and perspective projections using the keyboard. When the user presses the O key, the program switches to an orthographic projection, and when the P key is pressed, it returns to a perspective projection. This is managed through a boolean variable, bOrthographicProjection, which is toggled when O key is pressed. The PrepareSceneView function checks the value of this variable to set the appropriate projection matrix. Finally, this code is reusable in any project where orthographic projection is needed.