Model Loader Extension Report

This is a report for the SOFT356 extension to the model loader that was submitted for coursework 1.
Through the entire module I've struggled to get to grips with C++ and OpenGL but despite the intense struggle I managed to get the first coursework done and after discussing it with Swen I proposed extending the model loader as my second coursework. I struggled immensely with this submission as well and didn't manage to get a key feature done but I managed to add enough features to feel okay with submitting my extended model loader.

Setup and how to run

After downloading the folder and opening the solution the project will open and allow the user to see the file structure and load individual files. Opening the Source.cpp file shows the user the main file and methods used for the model loader. Source.cpp contains the file parser, display, light settings and camera controls for the model loader. There should not be any errors in the code unless the GLM and NupenGL packages need to be installed or re-installed for the project. The files are all present and after my original model loader I decided to change the shaders so that the original colour of the texture is displayed since I originally turned it purple.

Using the Model loader

After making sure there are no errors, all the media files are present and in the correct format (.obj) the user can press F5 or press the button at the top of the IDE that says "Local Windows Debugger" to run the program. After running they will be met with a command window where they will be prompted to enter the model they would like to load, currently the only model in the media folder is Creeper but when the user types that into the command window the display window will appear with the two objects loaded. The new implementation for the second coursework is the moving of the camera using the WASD keys and the mouse to change the rotation of the camera. This is using GLFW to read the inputs and act on them. The user can also close the window using the ESCAPE key but you still have to close the command window yourself.

Structure of the program

After taking inspiration from a mix of project 3 and project 5 the model loader is set out in a structure very similar to them and modified and moved.

As mentioned before the code for adding new features to the code from the previous submission. The lighting and shaders use the basic features from project 5 due to me still not quite understanding how to make my own.

The features listed below are all included in the project.

-GLM features used to render the cube
-Buffers and number of vertices used with GLfloat arrays
-Creating a camera and setting an initial position
-Setting time between frames
-Pushing data back into arrays based on the OBJ file
-Setting up the buffers to create triangles for the faces of the object
-Loading the texture through the MTL file
-Taking GLfloat values and loading information from the shader
-Configuring lighting and setting them up based on their type (Ambient and Specular)
-Drawing and displaying the two OBJ's with the appropriate texture
-Using a while loop for making sure the inputs are correctly managed
-All of GLFW inputs and being able to toggle between the wireframe and normal view
-Callback functions for mouse tracking and scroll wheel tracking

The features implemented for the second project are; Camera and movement, Setting up lighting, adding an extra cube and being able to turn off the texture to show the texture or the wireframe.

Evaluation
Throughout this module I have struggled to get to grips with the language and how to construct things using OpenGL. But using project 3 and project 5 from the sample code allowed me to kickstart the development of the coursework and that combined with various online tutorials I managed to get the coursework to a point where I am happy with it. With the help of tutorials I managed to integrate a system of lighting, the movement was inspired by a tutorial on LearnOpenGL that I found. If I was to develop the project further I would add a button to draw extra objects during the runtime.