# Computer Graphics Coursework – Self Assessment Document

**Name:** Kai Hodgson **ID number:** *23634157*

Complete the self-assessment grid below by writing a short explanation of how you have satisfied the requirement and how it has implemented in your code.

|  |  |  |
| --- | --- | --- |
| **Learning outcome** | **Mark** | **Weighted mark** |
| 1. Use appropriate mathematical tools (40%) |  | 0 |
| 2. Develop a 3D graphics application (30%) |  | 0 |
| 3. Write shader code (30%) |  | 0 |
|  | Total | 0 |

Your mark for each Learning Outcome (LO) is the highest mark achieved based on the criteria specified in the self-assessment grid. Note that you will need to have satisfied all criteria at the lower mark bands to be awarded marks in the higher mark bands, e.g., to get a mark in the 70 - 80 band for a learning outcome you will have needed to have satisfied all criteria in the 40 – 50 and 50 – 60 mark bands.

## Learning Outcomes:

**LO1** Select and use appropriate mathematical tools for constructing and manipulating geometry in 3D space.

**LO2** Develop an interactive 3D graphics application using an industry-standard API.

**LO3** Write shader code for the programmable pipeline on modern graphics hardware using an industry standard shader language.

## Self-assessment Grid

|  |  |  |
| --- | --- | --- |
| **Mark** | **Criterion** | **Comments (state how and where you have achieved the criterion)** |
| 42, 45, 48 | LO1: Basic use of vector and matrix objects | Vector and matrix objects have been used to allow for the assets to display at the correct positions. (Lines 147 – 173) |
| LO2: Application compiles and runs without alterations to the source code of CMake file. |  |
| LO3: Implementation of shaders to apply appropriate textures to objects. | Teapots have blue and diffuse textures whilst the floor plane uses the correct stone texture. (Lines 173-178) |
| 52, 55, 58 | LO1: Basic use of translation, rotation and scaling transformations. | Used to allow for visuals to display correctly to the user camera. (Camera.hpp file) |
| LO1: Implementation of glm library functions for calculating view and projection matrices. | Glm was used to ensure that matrices used the correct value and were calculated correctly. (Camera.hpp Lines 21-32) |
| LO2: 3D virtual world has been created using instances of a single object type. | As displayed in the screenshot above there are teapot assets, light sources, a floor plane and a moveable camera in my project. |
| LO3: Use of shaders to apply dynamic lighting from point light sources | The light file and all the 5 shader files were utilised to ensure that lighting was displayed as intended on the teapot assets from multiple light sources. |
| 62, 65, 68 | LO1: Implementation of students own functions for calculating view and projection matrices. |  |
| LO2: 3D world created using multiple object types. | I have utilised the teapot and the floor objects as displayed in (Lines 109-192) |
| LO2: Users can navigate the virtual world using keyboard and mouse inputs. | The program utilised a 3D camera that can be operated with keyboard and mouse inputs as displayed in the Camera.hpp and cpp files as well as the inputs being taken in at (Lines 193-204). |
| LO3: Use of shaders to apply dynamic lighting from different types of light sources. | Light Fragment, Light Vertex and Multiple Light Fragments files were all utilised to allow for more complex types of lighting from multiple light sources to be displayed over the teapots and look correctly from different angles. |
| 72 75, 78 | LO1: Implementation of students own functions to replace glm functions (e.g., glm::length(), glm::dot(), glm::cross() etc.). |  |
| LO1: Implementation of quaternions to calculate rotation matrix. |  |
| LO2: Interactive dynamic aspects of the virtual word and controllable by the user (e.g., position of objects, location and function of light sources etc.). |  |
| LO3: Appropriate implementation of normal and specular maps. |  |
| 85, 90, 100 | LO1: Use of quaternions to calculate view matrix. |  |
| LO1: Use of SLERP to smooth out changes in camera direction. |  |
| LO2: Implementation of a third person camera with the ability to switch between first and third period view. |  |
| LO2: The position of the camera or character obeys the constraints of the physical space (e.g., can’t pass through objects, can’t hover in midair etc.). |  |
| LO3: Use of shaders to apply parameter driven effects within the scene, e.g., light properties controlled using camera/character position. |  |