|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Description | 0 – 39% | 40 – 49% | 50 – 59% | 60 – 69% | 70 – 100% |
| Engine Code (including design patterns such as components and interfaces) (20%) | Program does not compile either through syntax errors or missing header or library files.  No or insufficient directory structure.  No or insufficient OO structure to the application.  No or insufficient number of new classes | Program compiles correctly on any machine (basic directory structure)  A minimal acceptable code structure | Good directory structure  No compilation or run-time errors  At least three new classes and one new component | A well-developed directory structure and engine  Evidence of code reuse and design patterns  Good use of components and game object types. | No fault can be found with the work in terms of design and code.  A well-developed structure using OO concepts and strong implementation of a component-based system and possibly other design patterns. |
| Data-driven architecture (including game-flow) (20%) | No use of data files | Basic use of files to load a single hard-coded file destination. Program has to be re-compiled to select a different file | Basic use of data files to load a scene.  No ability to reload the scene or load a different level | Several different levels have been created via data files.  The program has the ability to reload different levels as and when required. Limited usage of data files for other purposes  A ModelLoader class has been implemented, possibly with model duplication or other issues. | Use of JSON or XML files (and possibly small text config files) to establish a fully data-driven approach, with the ability to load/reload level data as well as other files e.g. configuration files, game object types, user controls etc.,  Data files control all flow of the game layer progression including when levels are switched.  A ModelLoader class has been implemented that only loads a model once and reuses when necessary |
| User Input (including cameras and player character) (20%) | No or seriously inadequate user interaction | Use of keyboard presses / mouse for simplistic movement of player character | Player-character can be moved around the scene.  No dynamic switching of cameras. | Good use of controls and the use of 2 cameras – potentially with minor issues | Strong use of mouse and keyboard for a variety of functionality.  At least 2 cameras fully implemented - first and third-person cameras attached to the player character. |
| Asset Pipeline – Serialisation and Parsing (20%) | All data files are written by hand | A simple scene of cubes/objects can be exported from Maya (with only their translation parameters) | A Maya scene with the ability to export objects. 2 GameObject types have been handled | Good use of Maya to build levels/scenes that can be easily added to a game. 3 GameObject types have been handled | Creative use of Maya and if needed custom programs to generate all of the data files needed, allowing for simple adjustments for all game and engine functionality. At least 4 GameObject types have been handled |
| Software Testing and Game Debugging Tools (10%) | No software testing has taken place | Extremely limited application of testing | Some testing and game-debugging has taken place | Good use of unit tests to try and ensure the code is working as expected.  The engine has some basic game-debugging capabilities such as text-to-screen. | Numerous unit tests have been created (and passed) to try and ensure the code is working as expected and can handle cases such as broken data files.  The engine has some basic game-debugging capabilities such as text-to-screen. |
| Professional Practices and Documentation (10%) | Inefficient documentation or use of professional practices | Readme included with controls/code usage.  Limited evidence of source control | Infrequent us of source control.  Documentation with some important information missing. | Regular use of source control  Documents contain all vital information | The assignment has been professionally handled with frequent and numerous source control commits over a number of weeks.  Evidence of good commenting and coding standards.  Documentation is well written and contains all vital information |