Requirements

**REPORT**

-professional formatting with clear headings and custom style implemented

-Relevant content, further explanation of mechanics, graphics, design patterns and logic. Appendecies used with Harvard referencing

-More than 3 referenced and titles images explaining some algorithms

**SOURCE CODE**

-Fully OO implementation. Excellent organisation (use of whitespace and indentation etc..)

-No warning and fully commented

-Consistent naming conventions

-No magic numbers

-Good use of constant values

-No global variables used

**VERSION CONTROL**

-Version control fully used, with ignore files implementation and evidence of 3+ weeks use and google drive

**VIDEO**

-Video supplied and fully demonstrates software

**GAME MECHANICS**

-Fully realises original specification, with all changes fully justified

-Game fully playable with no faults whatsoever

-Game fully tested. Most or all faults fixed. Where fixes not implemented, possible solutions presented with justified reasoning as to why this was not implemented

**NON-PLAYER ENTITIES**

-Entities fully interact with player (chase player in range, shoot player, run away from player, etc...)

-Advanced pathfinding implemented – explain technique researched and used

-Sphere collider, per-triangle collision with scene management (scene nodes/multi-part game objects) implemented

**INTERACTIVE**

-Separate input class implemented with direct input (keyboard and mouse/gamepad/both)

-Time class implemented with fps information correctly shown and delta time implemented for things like movement

**SCENE**

-Cubemap implemented for reflections on some objects

-1 or more additional cameras used (mini map, top down view, dynamic environment map)

-Can fully move around the map and interact with environment, also one extra action (jump, crawl)

-Advanced technique used for creating scene (explain technique researched and used). Full use of object pooling and instancing. Must use at least one object downloaded from the internet or self-made model.

-More advanced particles used (alpha/colour blending, textures, animated etc…)

-More advanced text used (custom font, alpha blended etc…)

**TRANSFORM**

-Transform with scale, rotation, translation correctly concatenated with matrix concatenation and fully utilising scene management and collision detection/ray casting (scene node)

**TEXTURING**

-More advanced textures used (bump maps, normal maps, dynamic environment maps, alpha/colour blended maps for text/particles etc…)

**LIGHTING**

-Multiple lights (more than one of each type of Phong Diffuse, point and spot)

**OPTIMISATION**

-Correct view/object clipping

-Object pooling

-No memory leaks/dangling pointers – show evidence

-BSP trees

**SHADERS**

-Pixel and Vertex shader used for lighting and other shader (Geometry) implemented

**OTHER FEATURES**

-Extra marks for (lens flare, god rays, bloom, depth of field, fog/rain shader effects, animations, dynamic environment map created for advanced reflections) -> choose 3 of them with 3 approachable points for each.

-collider class

-sphere collider

-box collider

-mesh (per-triangle) collider

-objfile model sphere collision

-geo model sphere collision

-plane collision

-triangle collision

-give objects tags like “ground”, ”enemy” etc.

-gravity until collided with ground objects

-create map

-build each upper platforms with functionality of hooking

-build mid platforms with collision

-build lower platforms with collision

-mechanics

-write algorithm for swinging

-write algorithm for air acceleration

-write cutting logic when player near other player

-write player shooting logic

-game logic

-create death timer and kill player when he has not hooked for a certain time

-create collectables

-create moving objects