**Learning Journal**

**Task 1 – Basic 3D Movement**

After finding that many online 3D movement tutorials were not quite what I was looking for, I decided to develop my own movement. I learned that I could put the movement Vector on one line if I added the x movement and z movement together. Then all I had to do was clamp the directional movement to 1 so that when I multiplied it by the speed variable it would never move at double speed, and set velocity to movement multiplied by speed after clamping.

**Task 2 – First Person Camera**

I learned that it was necessary to differentiate the x and y rotations. If I left the x rotation on the player then the player would be able to spin, so instead I had to just change the x position of the camera. I also had to have the player rotate on the y instead of the camera as if it was just on the camera then the player would not change the direction they were facing and my movement script would not work.

I also learned that I had to clamp the x axis rotation of the camera, otherwise the player would be able to move the mouse upwards and look behind them, viewing the world upside down.

**Task 3 – 3D Jumping**

I learned to use a layer mask to define what the raycast was hitting beneath the player. Using a layer mask allowed me to only register hits on a certain layer, meaning I could have other game objects in the scene that didn’t allow for jumping.

I also learned that I could not simply pass my direction vector through velocity as it would constantly set the y axis to 0, meaning that jumping would not work properly. I simply passed the x and z values of my direction vector through and eft the y value alone to fix this.

**Task 4 – Firing Projectiles**

I found that on instantiated a projectile, if the player moved then the projectile would as well because it was spawned parented to the player. To fix this I learned that you could instantiate prefabs as new game objects and edit them, so on spawn I set the game object parent to null. This also helped set a direction for the projectile as I could directly edit the Rigidbody after spawning a clone.

I also found that I had to create a destroy script on the projectile, as collisions can only be registered via OnTriggerEnter/OnCollisionEnter, etc. if the function is on the object with the collider.