**Summative 4 - Steering Behaviours**

**Seek**

My Seeking algorithm started with only a simple calculation that took in the Seekers position and the position of its target. The displacement from these positions was then travelled along by the Seeker for a small distance based on the Seekers speed and delta tick. This created an object that changed direction in an instant and would snap to a new target. To fix this issue I researched into turning speeds and maximum force to calculate a more realistic and slower turn. The resultant Seek function calculated the displacement of the Seekers and the Targets positions as before but this time the Seeker turned only a small amount ( up to its maximum turning speed) towards the new desired location.

Key Components:

Desired Velocity – The vector from the Seekers position to the targets position.

Maximum Speed – The maximum speed the Seeker can move in one second (Used in conjunction with delta tick).

Maximum Force – The maximum turning capabilities of the Seeker in one second (used in conjunction with delta tick).

Steering Force – The difference in desired velocity and the current velocity of the Seeker.

**Wander**

I created my Wander function based from Reynolds wander theory using a projected circle in front of the Object. The Object moved every delta tick towards a position on the circumference of the circle retaining the previous angle of the position so that the next tick the object would move in a similar direction as before. This gives a more realistic effect to the randomly generated path that the wandering object will take. Once wander has identified a new point on the circle to move towards, Seek is invoked to calculate the steering and velocity.

Key Components:

Circle Position – The position of the circle in front of the Object.

Circle Radius – The radius of the circle.

Wander Angle – The current angle of the targeted circle point.

Angle to change – The angle increment or decrement to the current angle to change where th object moves.

**Containment**

Containment is utilised to keep the objects from leaving the map/ terrain area. This effect was achieved by creating a smaller area within the terrain. This smaller area registered when an objects position left the containment area and would change the objects steering algorithm to seek the centre of the terrain and turn it around. The smaller containment field was implemented to give the Object room to manoeuvre and turn around without having to radically change its direction which looks more natural. Once the object is back within the borders of the containment field it resumes it original steering behaviour.

Key Components:

Containment Field – The smaller boundary within the main terrain boundary that was used as a threshold to turn objects around in.

**Flee**

Flee is implemented the same as seek but takes the opposite direction from the displacement of the Fleeing object and the target. Once that desired velocity is found then the same calculations are implemented where a turning speed and maximum speed are once again used to limit how far the fleeing object can move away and how much they can change their direction.

Key Components:

Desired Velocity – The vector from the Fleeing objects position to the targets position.

Maximum Speed – The maximum speed the Fleeing objects can move in one second (Used in conjunction with delta tick).

Maximum Force – The maximum turning capabilities of the Fleeing objects in one second (used in conjunction with delta tick).

Steering Force – The difference in desired velocity and the current velocity of the Fleeing objects.

**Evade**

Evade is an upgraded version of Flee where the Fleeing object determines the future position of the target and calculates the best path to get as far away from the target as possible. The distance into the future Is directly correlated to how far apart the two objects are with smaller distances calculating the future position much closer to the target and the larger the distance the further forward into the future the target is estimated. The estimation of the targets future position is based off the targets current position, speed and velocity.

Key Components:

Target Future Position – The estimated future position of the target.

**Flocking**

Flocking requires three rules to function correctly. Separation, Alignment and Cohesion.

**Separation** keeps the objects away from one another by slightly altering their velocity away from another when they start getting too close (generally defined by a radius around the object). This requires looping through all other objects to determine which ones are within their radius and need to be calculated. Once all objects have been checked an average is then calculated as the first steering force for flocking.

**Alignment** calculates the average direction of the flock mates to direct their own movement. This is also limited by a radius so not all flocking objects are used with in this calculation. This creates the second steering force for flocking.

**Cohesion** brings the flock closer together by calculating the centre position of the flock mates within the cohesion range and after taking an average dependent on how many objects were involved calculates the third steering force for flocking.

Once all three steering forces are calculated my Flocking algorithm also taking in a target position to seek towards ( in this case the closest player to the flock or the inidividuals).