# A Cure for Arachnophobia

## Implementations of Data Structures and Design Patterns:

### Main Scripts:

* The KD tree is implemented in Kdtree.cs (found in folder: Scripts\DesignPatterns\ Kdtree.cs)
* Observer abstract class: Scripts\DesignPatterns\Observer.cs
* Subject abstract class: Scripts\DesignPatterns\Subject.cs

### Soldiers and SightRange:

* The soldiers have a sphere collider that acts as their sight range. The soldier (Scripts\Soldiers\Soldier.cs) is an observer of the SoldierSight (Scripts\Soldiers\SoldierSight.cs – extends Subject).
* The Soldier maintains a KD tree of drones to figure out the nearest drone (which is what it targets and attacks)
* The SoldierSight script maintains a list of drones in sight.
* Whenever a drone enters the collider or leaves the collider, it notifies the soldier and pushes the updated list of drones in sight to the soldier script
* At this point, the Soldier updates its KDtree

### Groups of Soldiers move in formation:

* Implemented in Scripts\General\InputResolver.cs in the CalculateGroupMove() method
* When a group of soldiers are selected and given a move order, normally they would all head towards the one destination and run into eachother.
* This method calculates an array of destinations to assign to each soldier so that they end up in a formation around the destination point
* I calculate the average position of the soldiers selected.
* Then I use a KDtree to calculate the soldier closest to the average position.
* This soldier assumes the central position at the destination
* The other soldiers take up positions to the left or right of the central soldier, near the destination point.

### EnemyController and SwarmSpawner

* EnemyController is the Observer (Scripts\Enemies\EnemyController.cs)
* SwarmSpawners are the subjects (Scripts\Enemies\SwarmSpawner.cs)
* Each Drone has a Health script (Scripts\General\Health.cs)
* The EnemyController knows the max number of enemies that can be on the map on any given time and maintains the number of enemies on the map currently
* There are multiple SwarmSpawners on each map that continuously spawn enemies
* Whenever they spawn an enemy, they Notify the observer to increment its enemyCount
* Whenever an enemy dies, it notifies the EnemyController to decrement its enemyCount
* The EnemyController prevents the SpawnSpawners from spawning more enemies if the maxEnemies have been reached, until some die.

### EnemyController and SwarmSpawner Health

* The EnemyController is also an Observer of the Health script of each of the SwarmSpawner
* When a SwarmSpawner dies, it notifies the EnemyController of this.
* The EnemyController then removes that SwarmSpawner from it’s list of SwarmSpawners

### EnemyController and SoldierManager

* The EnemyController also Observers the SoldierManager (Scripts\Soldiers\SoldierManager.cs)
* The SoldierManager maintains a list of the soldiers and each of the soldier’s position.
* The EnemyController maintains a KDtree of the Soldiers positions
* Whenever any of the soldiers move, the SoldierManager notifies the EnemyController.
* The EnemyController then updates it’s KDtree with the new positions
* It then lets each of the SwarmSpawners know which the closest soldier is to the individual Spawner.
* The SwarmSpawner then updates it’s target to be the nearest soldier to it.
* However, the Drones that it has spawned normally just swarm around the SwarmSpawner.
* If the nearest soldier is within sight range of the SwarmSpawner, it let’s all its drones know to attack that soldier.
* If a soldier leaves sight range, the drones go back to just swarming around aimlessly.