SE2250b Assignment 1: Part A
Boluwatife Ademiudn 25109004

One of the requirements for this game is that all pick up objects are created dynamically. The method I used was to this was to instantiate multiple instances of different prefabs in the Start method of the player controls script. In my game I had 3 different prefabs, each prefab was a different Game Object with a different number of points earned when triggered. I wanted 5 of one Prefab, 4 of another prefab and 2 of the last prefab. In order to do this, I created a public Game Object variable for each of the three prefabs, then I made copies of each prefab one by one using instantiation, each copy is a pickup object. Doing this meant I manually typed out the vector position of each pick up object. An advantage of spawning the pickups this way is it I can ensure each pickups spawn at a different vector location and that no pickups overlap. A disadvantage is that I had to manually type out the code for each GameObject I instantiated, and this was slightly repetitive and time consuming. An alternative method would be to create a for loop for each prefab object. Since I had three prefabs, I would have 3 for loops. Each for loop would instantiate copies of each prefab a certain number of times, while picking a random location for the vector location each time by using the Random.Range function. An advantage to this is that it limits the amount of repeated code and it brings game variation as pick-ups can appear at a new location each run. A disadvantage is that the addition of the for loops would make the script take slightly longer to compile.

Another requirement is for the game to pause then restart when there are no more pickup objects to collect. The method I used was to create a Restart method which held the SceneManager.LoadScene function which was executed when Restart was called. I sent the name of my scene (which was Minigame) as a parameter through the Manager.LoadScene function. Then, I made an if statement in my onTriggerEnter function that would call the Restart function if the maximum score was reached (meaning all pick up objects were collected). An advantage to his is that the code is relatively straightforward and easy to implement. A disadvantage is that if I were to change the name of the scene in Unity, I would have to open up this script and manually change the parameter passed into the Manager.LoadScene function. An alternative method would be to pass the SceneManager.GetActiveScene().name function as a parameter of the Manager.LoadScene function. The if statement in the onTriggerEnter function would stay the same. An advantage to this is that I would not need to manually change the name of the parameter in the script if I changed the scene name in Unity. A disadvantage is that this method Is slightly less straightforward as it requires the use of an additional function.