~\OneDrive - St Paul's Catholic College\Documents\2D Strategy Game - Liberator\Assets\Scripts\Hexes\Distance.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using TMPro;
4  using UnityEngine;
5  using UnityEngine.UI;
6
7  public class Distance : MonoBehaviour
8  {
9      public int distanceFromStartingPoint; // counts distance from the starting hex
10     public int stepsToGo; // determines the number of steps to go through the hex
11     HexData hex;
12     TextMeshProUGUI distanceText; // refers to to the text component of the same object
13     private void Start()
14     {
15         hex = GetComponentInParent<HexData>();
16         distanceText = GetComponent<TextMeshProUGUI>();
17     }
18
19     // sets distance from starting hex and displays it
20     public void SetDistanceForSoldier(HexData initialHex)
21     {
22         // add a step to the previous step to get distance from starting point
23         distanceFromStartingPoint = initialHex.distanceText.distanceFromStartingPoint +
initialHex.distanceText.stepsToGo;
24         // display new value of the distanceFromStartingPoint
25         DisplayDistanceText();
26     }
27
28     private void DisplayDistanceText()
29     {
30         distanceText.text = distanceFromStartingPoint.ToString();
31         distanceText.color = new Color32(255, 255, 255, 255);
32     }
33
34     public bool EvaluateDistance(HexData initialHex) // compares distances between two
hexes
35     {
36         return distanceFromStartingPoint + stepsToGo ==
initialHex.distanceText.distanceFromStartingPoint;
37     }
38
39     public int MakeMePartOfOptimalPath() // includes this hex into optimal path list,
returns number of steps to go through the hex
40     {
41         OptimalPath.optimalPath.Add(hex);
42         hex.Landscape.color = new Color32(150, 150, 150, 225);
43         return stepsToGo;
44     }
45
46     public bool EvaluateDistanceForSoldier(HexData initialHex)
47     {
48         // distance to reach initial hex and get out of it
49         int currentDistance = initialHex.distanceText.distanceFromStartingPoint +
initialHex.distanceText.stepsToGo;
50         int stepsLimit = Controller.soldier.steps; // velocity of a hero
51         // default value of distanceFromstartingPoint is 20 to set the shortest path
52         return distanceFromStartingPoint > currentDistance && stepsLimit >=
currentDistance; // to evaluate if the velocity is enough to reach this hex
```

```
    }
    // Update is called once per frame
    void Update()
    {

    }
}

```