Final Evaluation Report

Advanced Data Structures

Student Name: Gianni Coladonato, 2414537

Date: 02-05-2025

Introduction

In this report, I will cover and reflect upon the data structures and algorithms used, challenges I faced, and design decisions made. The game I chose to work on for this assignment was the game I created for my last semester (Game Engine I), SpellBlade, since I knew that there was a lot that could be added/improved.

Week / Tier 1

In the first week, I was tasked with creating an Array or List in the game. I decided on implementing a List of Potion objects. These potions would hover and behave like regular items in the game, except when picked up they apply a temporary, 5-second boost to a player's stat (Speed, Jump Power, Damage, All the Above). These potions would be stored in a List inside of a GameObject called level manager, and the level manager would decide to spawn one of these potions at random (randomly generate a number corresponding to the index positions of the potion list) at a specific point in a level/scene.

Challenges (Week / Tier 1)

I found this implementation to be fun, as I already had experience Coroutines, which I use frequently in my Game Engine II course. Whenever a potion applies it's effects, it starts a coroutine that removes said effect after 5 seconds, simple and clean and fun to code. One challenge I did have with this though, was changing how the player's damage was calculated, as before it was a set variable that the Player Controller could not modify. I had to write a script to access the Attack scripts of the player's attacks, called PlayerDamage, which became extremely useful for the rest of the assignment. During the assignments for my GameEngine II class, I learned the power and usefulness of Singletons, something I wish I used when making this game for the first time, as it's useful when accessing single objects in the game scene such as the player, a level manager, canvas manager or sound manager.

Week / Tier 2

In the second week, I was tasked with creating a Linked List or Hash-Table in the game. I decided on implementing a Linked List of Weapon objects. These weapons would have different properties (Damage and Knockback) than the player's base weapon, and the player can freely swap between them by pressing Q or E. The top of the screen shows the player which weapon they currently have equipped.

Challenges (Week / Tier 2)

Thanks to the work done in the previous week, this step became fun and satisfying to complete. I implemented code in the player controller to change the player's current weapon to whichever was in the list next or previous (with code to wrap around to the start if the player was at the end of the list), the code would then pass the weapon class to the script PlayerDamage and change the damage and knockback of the weapons accordingly.

I implemented two weapons, the sharp sword, a weapon that deals high damage but low knockback, and the heavy sword, a weapon that deals lower damage but applies a greater knockback. Both weapons have their pros and cons, depending on a player's playstyle, they may prefer to always keep enemies away, or try to knock them off ledges (heavy sword). Or the player may prefer to dispose of enemies as quickly as possible despite the risk of taking damage (sharp sword)

Week / Tier 3

In the third week, I was tasked with creating a Tree or Graph in the game. After being stuck on what to implement, I decided to implement a Tree data structure, specifically, a Tree of Upgrade Nodes, creating an Upgrade Tree in the game. Throughout the game, the player can find upgrade points, that when collected, unlock either a left or right node (Upgrade Node), when unlocked, the player receives a minor stat boost based on the Node's name (Speed, Power, HP or Jump).

Challenges (Week / Tier 3)

I was initially anxious on this week, as I wasn't sure where/how to implement a tree at first, but once I got the idea and tried it out, I found the Tree data structure to be easy to implement and navigate, like a linked list of sorts. For this tier, I ended up using parts of code from both previous tiers of the assignment, the LevelManager, to spawn upgrade points throughout specific parts of specific levels, the PlayerDamage and logic like PowerUps being used to apply these buffs to the player, and the CanvasManager being used to create a visual representation of a tree. The hardest part of this tier was figuring out how to fix the spacing of nodes, since the children nodes of the left and right branches would often overlap visually. This was solved by tracking the number of children of a node and adjusting the width accordingly.

Conclusion

During this assignment, I learnt the power, usefulness, and potential applications that data structures like Linked Lists and Trees (Trees being surprisingly easy to store/save) have and have already began utilising them in other coding projects in my Game Engine II Final Project.