Quarter 2 Prism

Juan Garza

# Summary

Throughout the quarter, I personally contributed to the game’s development by implementing Walls on the map, collision in walls, spawn player in map, spawn enemies in map and spawning objects at random in map. Multiple scripts using the previous algorithms were implemented to ensure that the random pre fabs placed on the map were randomly generated and not in the way of the corridor, preventing passage to the player to the next room.

# Algorithms and scripts

## **Prefab placer and Room content Generator.**

Our team together with our mentor engaged in a method of approach in which we should continue the dungeon. In quarter one, we focused on the first room generator, and corridor algorithm, the natural step was to ensure the walls of the dungeon and corridors were places along with a collision with the player. Both algorithms serve so that the items are placed on the map.

### *Descriptions*

Our team engaged in extensive discussions regarding how we wanted the game to shape up regarding its map and area. We wanted to make sure that the dungeon was as organic and natural looking as possible. We decided on this approach as it was the one with what we had in mind. I personally contributed to the addition of the random player spawn, items and enemies. The item placement prefab placer script ensured that the items were not only places randomly on the map, but it would track the corridors, ensuring that the items would never be in a corridor path to avoid the player getting stuck in the room. The player spawns in the dungeon at random, along with random enemies.

### *Scenarios*

A picture containing histogram

Description automatically generated

Since it was important that the items places were not in the way of the corridors, we implemented an algorithm to keep track of them. This also showcases walls with collision included. We collected this data to ensure that.

A screenshot of a video game

Description automatically generated with medium confidence

We first tested this in our first room, we saw that the items were placed correctly. We for now have a lo-level room system with just dungeons as we have not decided on each specific room purpose. That is an idea for future iterations.



The player being places in the dungeon with items alongside the player. The items as you can see are placed ether near a wall or in the room but never in the path of the corridor.

Graphical user interface

Description automatically generated

Using the same principal for the random prefab placement, we were able to implement that for placing random enemies. future iterations will hopefully control the population of them in each dungeon.

# Task

Jira was utilized by the team to create tasks and sprints for scenarios, enabling us to complete the foundational elements of the game in a week. This strategic approach was designed to avoid any potential delays in future tasks that may rely on preceding ones.

I worked with my teammates to create a sprint in Jira, and helped decide who would do each task. I focused on the environment of the game from the map, items, corridors and its placement along with walls and tiles.

# Code

I followed a 2D prefab placement along with items and enemies on YouTube on how to implement each algorithm to our game. The tutorial exampled the way to ensure that the corridors and dungeons were going to be connected by the end of the tutorial. I was having a hard time updating my github using my computer my teammate was able to do so but the link will lead you to the submission.

Used the following youtube videos in the process

Prefab placements:

<https://youtu.be/t1a1QBcfRlM>

enemy placement:

https://youtu.be/JOkOih1AKsQ

Wall placement collisions:

<https://youtu.be/-QOCX6SVFsk>

<https://github.com/UTRGV-CS-Projects/202320-spring-2023-projects-prism-bossrush/commit/c8919dd7dfb88464d42dffae309212fde8a7c822>

 Algorithms showcasing Placing enemies in the dungeon.

Text

Description automatically generated

Algorithm showcasing item placement as well as creating that item in it.