Gillian Croteau

4-27-22

TP2 Project Proposal

Project Description

My term project is a game called Sands of Time. In this game, you enter a dungeon with the goal of escaping with as much gold as possible. In the central room is a large hourglass that is counting down time, and an exit portal. There are several hallways branching off. You must navigate the dungeon and collect gold. However, there are mazes and puzzles to navigate and mummies that will attack you. If the mummies deplete your health, you die and lose. You can also find piles of sand hidden in the dungeon which can be brought back to the main room to add to the hourglass and increase your available time in the dungeon. You can leave at any time through the exit portal, at which time you win whatever gold you have collected. However, if the timer runs out, the portal closes and you get trapped and die. The better rewards are further in the dungeon, but you run the risk of running out of time if you get lost.

Competitive Analysis

This game is based off of the Minecraft Championship game, Sands of Time. Minecraft Championship is a tournament held several times a year between a handful of large Minecraft Youtubers, streamers, and other content creators. Eight games are played per tournament out of a larger selection in teams of 4, and Sands of Time is one of those possible games. In Sands of Time, the teams must explore a dungeon and try to escape with as many points as possible. If they are locked in the dungeon when the hourglass runs out, they get no points for that game,

which can be detrimental to the final team score and almost guarantees a team will not win first overall. The basic mechanics are similar to my version, where there are many rooms in the dungeon, including mazes, puzzles, and rooms filled with assorted Minecraft mobs. They must collect coins and bring sand back to the central hourglass. To learn more about Sands of Time visit this link: https://mcchampionship.fandom.com/wiki/Sands_of_Time

The challenges with adapting this game for a term project start with the fact that this is a team game and my version is single player. The strategy is very different. One place where this comes into play is the timekeeping mechanics. In the Minecraft version, there is no central clock ticking down the seconds, only an hourglass. Often teams will appoint someone to be a timekeeper and watch the hourglass to keep their team updated on the time. However, this is a single player version, and I did not want players to be disadvantaged by not knowing their time deeper in the dungeon. I added the clock, which is always displayed, to combat this. In addition, in the team version, a player death does not result in immediate ending of the game. They lose all of their gold and become trapped in a cage, which their teammates must release them from, after which they can continue to play the game. Since this is single player mode, there is no way for a teammate to release the player and player death results in game over. The second largest challenge is how to play the game. The original game is played in Minecraft, which is played in a first person perspective in a 3D world. However, I did not want to implement my game this way, so this game is played in a top down style, similar to Stardew Valley. The planned graphics are also inspired by Stardew Valley. Removing the Minecraft environment also presents some challenges, as the monsters within the dungeon must function differently than traditional Minecraft mobs.

Structural Plan

In the project, the main file stores all of the graphics and runs the game. There are several additional files. One stores all the functions needed for the higher-complexity elements, such as pathfinding and maze generation. One stores all of the classes and methods associated with them, such as the Mummy class with the pathfinding method. The rooms are also an object, as well as individual piles of sand. One file stores the function used to initialize the rooms and set up the dungeon map.

Algorithmic Plan

The most algorithmically complex parts of this project are the maze generation for the maze rooms and the pathfinding algorithm for the monsters. To do the pathfinding, I chose to use Breadth First Search for ease of finding the shortest path from the monster to the player. Since the rooms are all represented as graphs, the BFS algorithm works in the wide open rooms as well as more maze-like rooms. This algorithm is easy to implement as it loops through a queue to visit all neighbors to check for the target. This generates a map of the path from the start to the end point. As my map was stored in another dictionary, I used a recursive helper function to extract the list of nodes in order. I used the TA Pathfinding guide to help me write the BFS algorithm.

To generate the maze rooms, I chose to use Kruskal's algorithm. To implement this algorithm, I started with a fully connected graph of the size of the room I wanted, and an empty graph. I created a randomized list of all the edges in the maze, then looped through them to check if they were connected. If they were not connected, I added a new connection in the empty graph, and if they were, left them alone. I utilized my BFS function to quickly check if the nodes were

already connected. I used the TA maze guide and “The Buckblog” website to help me write this algorithm.

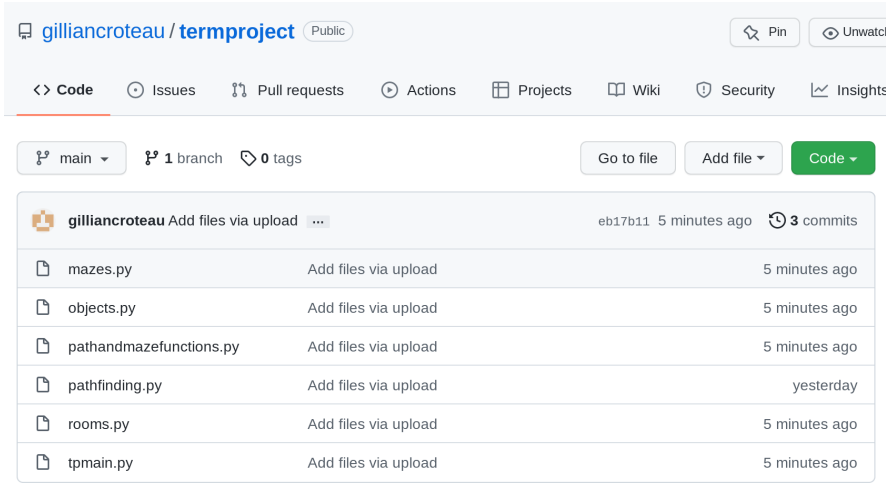
There will also be an algorithm for generating the random dungeon. There is going to be a preset list of rooms that must be arranged into a 2D list to use a map. This will likely not use a traditional maze algorithm, as it is under more constraints. For example, the center room must always be the same. However, it may still utilize the BFS algorithm to check to make sure all the rooms are actually connected after being placed into the map.

Timeline Plan

I intend to complete the randomized dungeon map, many more room types, and assorted big fixes by TP2. Since I also plan on using pixel art for the rooms and sprites, I plan to start adding this by TP2 with it finished by TP3. By TP3, I plan to add more features to the dungeons, like puzzles and highly lucrative but dangerous vaults. For vaults, you must find a key hidden somewhere else in the dungeon to open them.

Version Control Plan

I am using GitHub to back up my project. I upload my changes files at the end of every work session with notes on what was added.



Module List

There are no planned modules at this time.

TP2 Update

For TP2, a variety of new rooms have been added to make the dungeon larger. The layout of the dungeon is still hardcoded, but an algorithm to randomize the layout is planned for TP3. The rooms have many different sizes and features. To implement this, the code to run the doors had to be updated as well as the new rooms initialized. There has also been a very basic help screen added on the start menu. In addition to the randomized dungeon, the plans for TP3 include improved graphics, some gameplay updates and new features, and bug fixes.

TP3 Update

For TP3, there were several new features added. The graphics were redone, including sprites using the PIL module and new room backgrounds. The dungeon is also now randomized, so rooms will shift around during every new play. In addition, there is now a login system to save high scores. This is done through the terminal, where the player is prompted to input a username. If this username exists in the system, the previous high score will be displayed. If not, a new account will be made. At the end of the game, the new score will be saved to the account in a json file. One notable bug fix was fixing the ability to impale yourself on maze walls.