Project Name: Turtle's Way Home

Project Description:
This project is a game with the main objective being making sure the user controlled turtle character is able to solve the randomly generated maze. There will be different difficulty levels for the mazes the turtle needs to get through. Every time the game is restarted, another random maze will be generated. There will also be a set timer that keeps track of how long it takes the turtle to get from one end of the maze to the other end. The turtle will also have 3 lives. Throughout the maze, there will be plastic floating around and sharks floating around. The plastic will move randomly and change speed. Everytime the turtle hits the plastic, it will lose one life. On the other hand there will be a shark that will know the path home and will be swimming that way, but not as fast as the turtle. If the turtle hits the shark, it will lose 2 lives. If all three lives are lost, the turtle will be respawned at the top of the maze.

Competitive Analysis:
As maze games are very common, there are many games similar to the one I am going to create. Pacman is similar as the user is trying to eat all the fruit without being hit by the ghosts in the randomly generated maze. There are also other simpler maze games where the game starts with a simple maze. There is no time limit and there is nothing chasing after the user. However, the user still has to get to the target location and every time it gets there, they can unlock a new level.

My maze game will be different from the simpler versions of the game as there will be certain objects that my user has to avoid when they are trying to get from one end to the other end. There are also lives that the user has to maintain and there is a time constraint the user has to beat, so I think the different challenges that are added on top of trying to solve the maze make it different. My project is similar to the other ones as a random maze will always be generated and that is similar to pacman. However, unlike pacman, the user does not have to eat all the food, rather the user simply has to make it to the end.

Structural Plan:

I will have many different classes for the different items in my game. I will have one class that contains all the attributes of the maze. I will make it a struct therefore I can continue adding to the class outside of the class definition. The maze attributes will include the number of rows, the number of columns, the height and width, the blank generated maze, the solution path, the path the player chooses, the player location, and so on. I will also have another class for the different locations that the player can be. Once again, I will define it so that I can define the attributes outside of the class definition.

Apart from the different parts of the maze, I will have a class for the turtle, plastic, and sharks. The turtle class will contain the user's position, and it will contain the image that makes the turtle. It will also contain the number of lives that the turtle has. It will always start at 3 every time

the turtle has to be respawned to its original position. However, everytime, it ends up on the same coordinates as either the plastic or the shark, one of the lives will be lost.

For the plastic, it will also contain the positive of the plastic. Since the plastic moves at a random current, the class will contain the list of the random speeds and directions that the plastic can move. In the class, there will be four different movement/wraparound functions. The first one will make the plastic move from right to left. The second one will make the plastic move from left to right. The third one will make it move from the right top corner to the bottom left corner. The fourth one will make it move from the bottom right corner to the top left corner. All of these functions will make the plastic wrap back to the top once it reaches the end of the screen.

The shark will also contain the x and y coordinates. The shark can only move in a certain direction and at a certain speed. Therefore, the speed will also be contained in the class. Since there are different levels, there will be a list of levels because the shark will move based on the level of the game chosen. The maze will be solved and the list of coordinates that are included in the solution will be outputted. This list will be contained in the shark class so that we know which specific coordinates the shark will be moving in.

I will also be using the best template for each level of the maze because I want to keep track of the best time that each player has for each of the mazes.

In the entire app started, I will call all the different classes so that all the different animations are created. In the timer fired, I will make sure the seconds elapsed are stored as a timer is a part of the game.

Algorithmic Plan:
I believe that the most algorithmically challenging part of my project will be the random maze generation. In order to solve the random maze so that I can have an enemy(the shark) moving on this path, I will use either Prim's algorithm or Kruskal's Algorithm. For the movement of the plastic, I want to be able to find all possible paths from one end of the maze to the other. They do not have to be the fastest path, they just have to be a path. Therefore, I will be using DFS and backtracking to find them.

When it comes to the user interface and making the turtle move from one end to the other, I will be using the CMU 112 graphics. The user will need to use the different arrow keys and different letters when they want to do things like choose the difficulty level or when they want to restart the game.

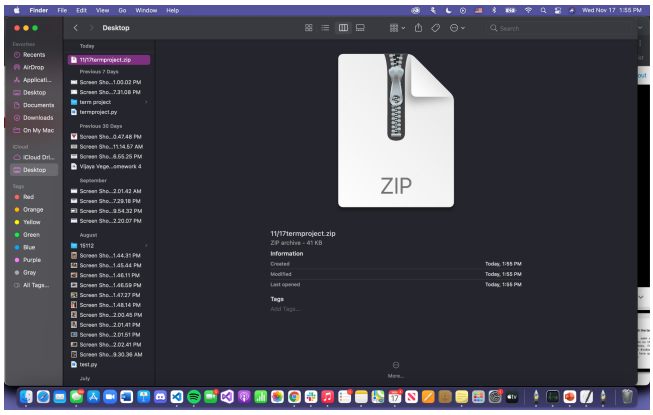
For a lot of the visuals, I will be importing images to use.

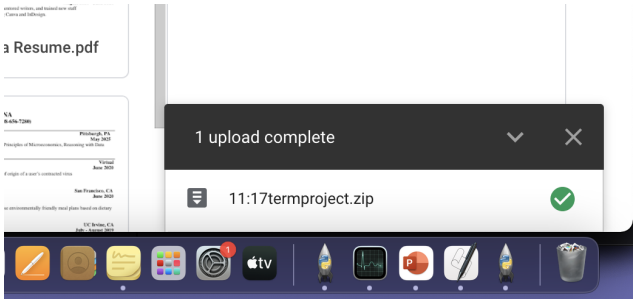
Timeline Plan:
By TP1, I will have a random maze generation, a turtle that can move by the user's controls, the number of lives displayed, and a timer at the top of the screen.

By the end of the weekend, I will make sure that the turtle can only move in the maze, have different levels of the maze, and create moving plastic and a shark. I will also have the

By TP2, the shark and plastic will be moving in their designated paths and whenever they hit a turtle, one life of the turtle will be lost. If the turtle reaches 0 lives, I will reset the game but keep the timer going.

Version Control Plan:
I am going to be storing my code on Google Drive. I will zip my code every night and upload that version to my google drive. I will label each file with the day's date and that will help me keep track of the specific code that I need.





Module List:
None

TP 2 Update:

- To reach MVP, there will be a new enemy class
    - This enemy class only moves when the user is in that enemy's quadrant
    - The enemy class will randomly generate a graph and then use BFS to solve it
    - The solved path will return a list and then that list will be used to give the enemy coordinates to move
    - If the turtle hits this enemy class, they will lose a life

Notes from User-Study-A Thon

- Implement a Start screen where users can choose which level they want to play
- Change the speeds of the red dots so they are moving less fast and its easier to actually win the game
- Change the speed of the purple dot because it goes too fast it is almost impossible to see the path that it is moving
- Try implementing different shapes for the enemies
- Try implementing a different number of dots for each of the levels that are chose
    - Will make the game look better, less crowded, and easier to play
- Make the maze darker

TP 3 Update

- There is a new enemy class which is a big fish
- This new enemy class will randomly generate a graph each time the game is run and then solve the graph using BFS
- This solved path will return a list of rows and columns and this list will the be converted into coordinates for the fish enemy to move
- The enemy class will also move in different ways depending on which of the four quadrants of the entire maze the turtle is in
- If the turtle hits one of these enemies, it will lose one life
- The plastic trash enemies start at random locations and go at random speeds each time the game is run
- Pressing s gives the solution to the maze which is found from DFS
- There are 3 levels to the game and each level creates a more complicated maze
- The game will go to an end screen once the user either solves the maze or runs out of lives
- There is a home screen which displays the rules, the controls, the username and the level chosen
- There is a marketplace screen
    - This screen displays how many points there are
    - This screen also displays the current number of lives
    - The bottom of the screen has three different options of how many lives you can buy and home much these lives cost
    - If you don't have enough points to buy a certain number of lives, it will displaying a message saying so
- If the user has remaining lives when they solve the maze, they will receive points
    - These points can then be used in a store where the user can buy some additional lives
- Based on which level of the game the user is playing, there will be a different number of enemies and the enemies will be moving at different speeds
- There is a leaderboard
    - Everytime the game is started, it will ask for a user name
    - Using a text file
    - When the end game screen is displayed, it will also display the user name with the best time and that time
- There are also images replacing the circles which were originally both the turtle and the enemies