**Design Proposal - Tiger Huo, 15112**

**TP3 Updates:**

- Socket successfully implement

- Sounds & sprites added

- New game mode: bombing (multiple player)

**TP2 Updates:**

- Start game screen

- Setting UI + instructions

- Updated Spider model

- New idea: Using socket to make a multi-player game

**Project Proposal:**

**Project description:**

Game: 'Spider: No Way Home'

A maze game in which you need to help the spider escape from a maze. You can't control the spider, but you can rotate the maze and let the spider fall. There will also be lock & key in the maze. 9 randomized maze algorithms implemented, and a path finder helper that can be enabled.

**Competitive Analysis:**

This game is inspired by the game Teeter on phone. The similar core feature is that only the map can be controlled, not the character. The difference is that Teeter focus on tilting the phone and control the ball on a horizontal plane, while this game focus on rotating the maze and control the spider on a vertical plane. There are also different features that Teeter doesn't have in this game, such as lock & key in the maze.

**Structural Plan:**

The final project will be separated in following python files:

- main: build up the CMU graphics structure and run the program

- rotateMaze: anything related to generating the maze, rotating the maze, path finder, lock & key, and displaying the above feature

- spider: a spider class that includes calculate speed and rotation of spider, collision detect, pull back & rebounce algorithm, and move with wall if collided.

- anime: anything related to animation.

if more features are enabled in the future, such as 2.5D or pseudo 3D, separate file will be used for calculating transformation.

**Algorithmic Plan:**

Maze will be the major part of the complex algorithms. 9 maze algorithms, including Recursive Backtracker, Kruskal's Algorithm, Prim's Algorithm, Wilson's Algorithm, Hunt-and-Kill Algorithm, Eller's Algorithm, Recursive Division, Binary Tree and Sidewinder Algorithm are included. They are all, except for binary tree, using recursive way to generate maze. There is also a maze solver using backtracking(with just in one line).

In one of the algorithm for pull back the spider if it hit into the wall, a binary tree method is used to maximize the efficiency.

In the future, 2.5D and pseudo 3D may, but not guaranteed to, be implemented and also contribute to the algorithm.

**Timeline Plan:**

Until TP1: complete all feature mentioned and debug motion calculation part

Until TP2: improve game UI and graphics, exploring 2.5D/ pseudo 3D graphics

Until TP3: implement 2.5D/pseudo 3D graphics

**Version Control Plan:**

Every time a large change is implement, the code will be uploaded on Github. Past versions can be founded in code history.

Figure 1. All files are uploaded on Github.

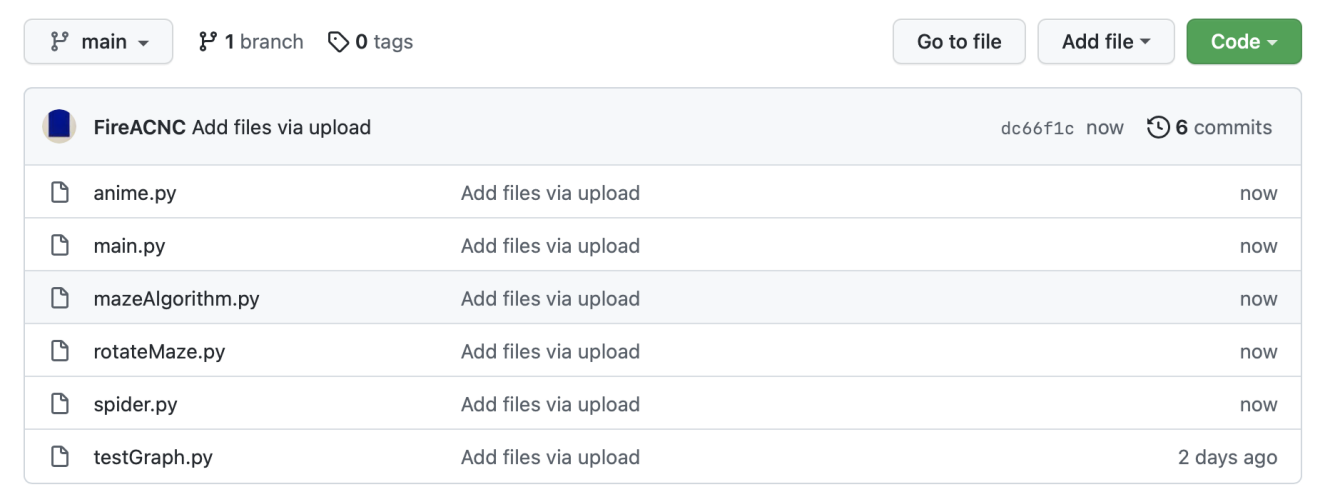
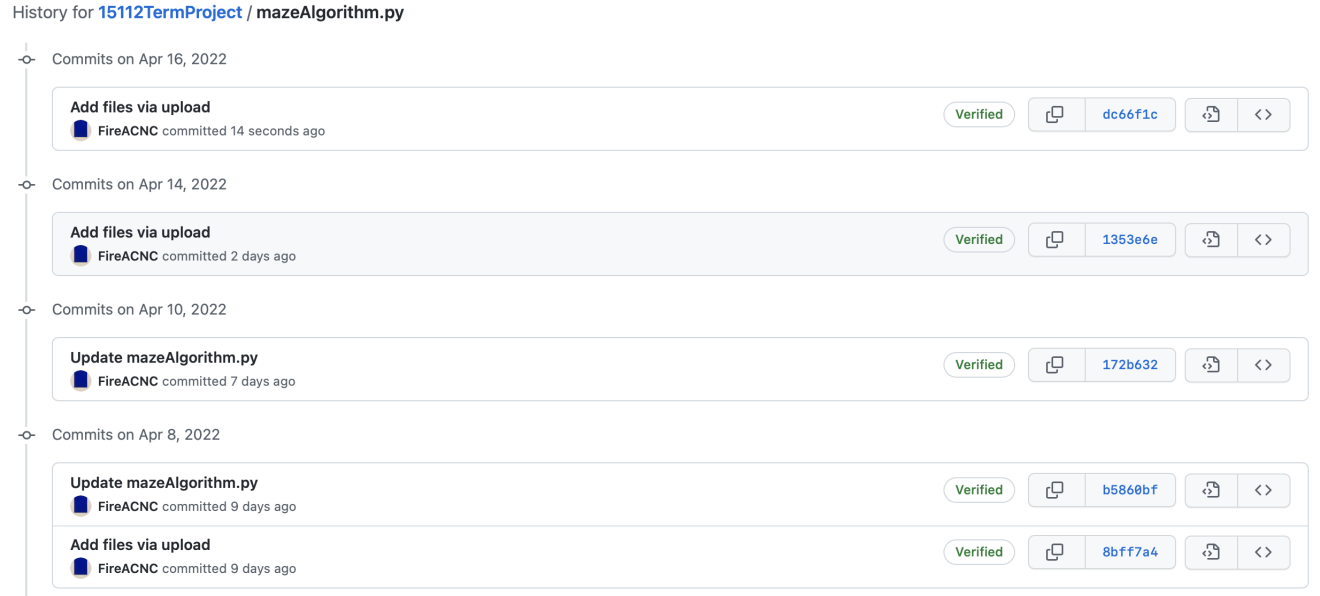


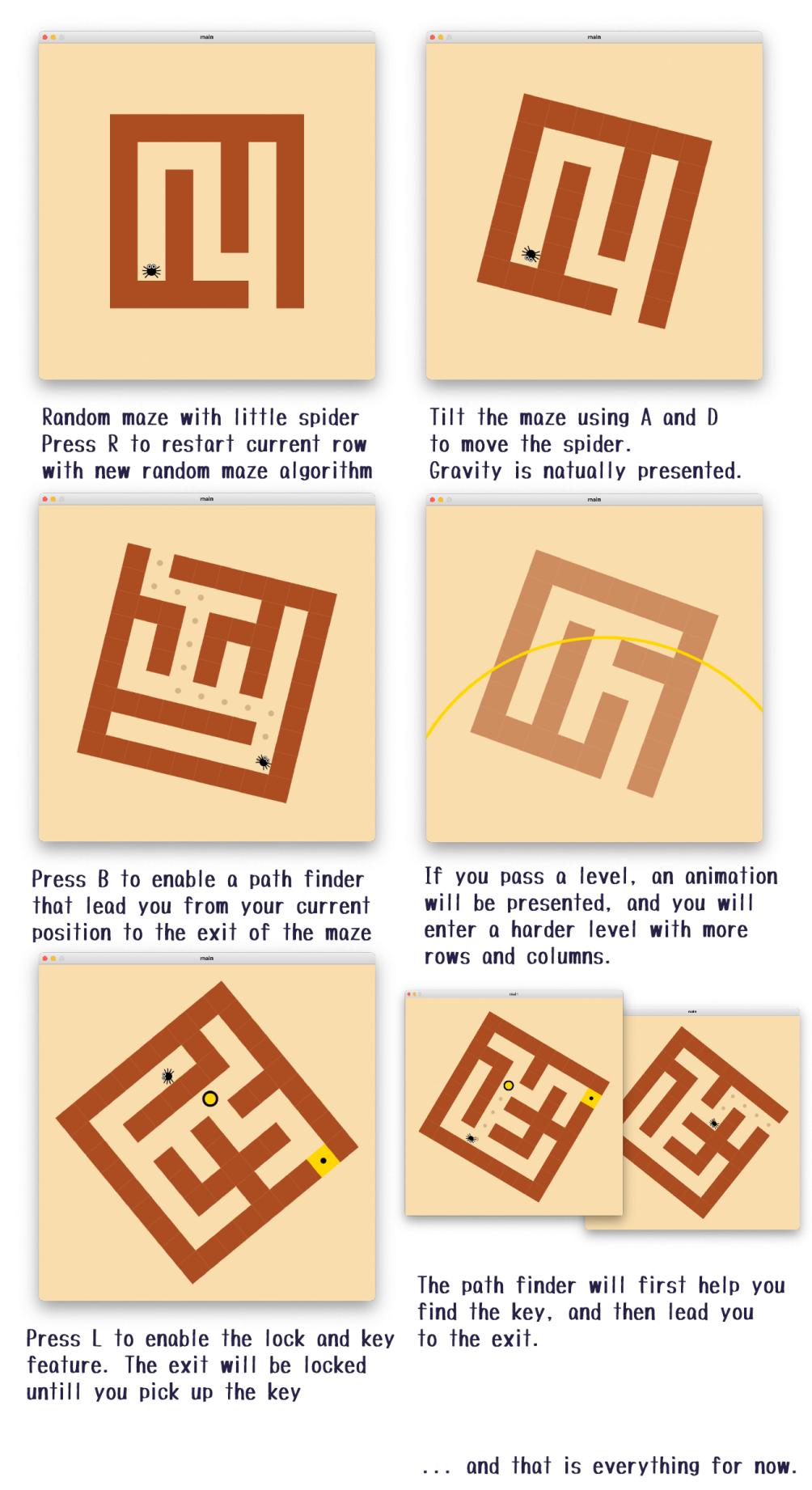
Figure 2. Code history can be tracked back to anytime.



**Module List:**

No external module will be used.

**Storyboard:**

****