Software Implementation and Testing Document

For

Group <12>

Version 1.0

Authors:
Abigail Centers
David Song
Paul Santora
Jason Hamilton

1. Programming Languages

We are using the Unity engine to design the layout and overall functionality of the game. We are also using C# for scripting triggers, character/enemy movement, events, etc. The Unity engine is a powerful game development platform with an extensive library of tutorials, start-up applications, and readily available sprites. These tools will make game development fun and easier to understand.

2. Platforms, APIs, Databases, and other technologies used

We will use the Unity platform (C#) to build the maze game. Unity already provides the necessary tools for creating a leaderboard and hosting the game online. We will be using git for version control for our project and Visual Studio as an IDE to write our C# scripts.

3. Execution-based Functional Testing

Bugs

1) improper asset imports
2) player avatar getting stuck on walls
3) scripts getting detached
4) when going back to main menu, game time would pause
5) menu animation/navigation issues

Performed most testing via Unity's built in play test and Visual Studio Debugger. We play tested the game thoroughly for bugs by testing for unexpected behaviour and edge cases (ex. moving the character to every position in the maze, collecting all items, comparing expected score/game clock to actual output on the UI).

4. Execution-based Non-Functional Testing

Play testing for memory leaks (by running the game for an extended period of time and checking system resource usage), correct animations (without choppy movement and appropriate number of frames), smooth transitions between scenes, user friendly menu navigations (minimal and descriptive buttons), and crashes due to script errors and warnings.

5. Non-Execution-based Testing

Ensured that project had version compatibility with different operating systems, MacOSX, and Windows, and between Unity versions. We did weekly code reviews, walkthroughs, and inspections of our gameplay. Checked that certain scripts and assets were attached to the correct objects. We reviewed our code over discord calls when teammates had conflicting schedules.