1 Game

1.1 Description

The game will be a 2D arcade-style game à la Pac-Man. The user will play as an Easter Bunny, who wants to collect eggs for the upcoming Easter. The setting is a hedge maze; upon collecting the necessary eggs, a door will open allowing the user to exit the maze and win the game. The Bunny also has a health value, if that value becomes non-positive, the game will be lost.

There will be traps scattered around the maze to harm the Bunny. Such traps *may* be hidden from view until they are activated and can do things such as trap the Bunny for a few seconds or do damage to them. Inversely, there will also be positive "traps" that can heal the Bunny, make them temporarily invulnerable, or give a bonus to the final score.

There will also be several hunters trying to hunt the Bunny; if the Bunny runs into a hunter then the game will be lost. Some hunters are also trappers, meaning they have the ability to drop harmful traps.

The final score will be shown once the game ends and will have many factors such as the time taken to finish, the final health value, and the bonus rewards collected.

1.2 Additional Notes

Listed are some additional customizations that may be added:

- The maze and all its contents will be procedurally generated.
- There will be several difficulty levels, which will determine everything from the number of required rewards to the starting health value.
- The hunters will be have some sort of artificial intelligence which allows them to chase the Bunny in a realistic manner.

2 Design

The main object will be the "game" object, which contains the maze and the user interface.

The maze will be represented as a 2D array of "maze objects". These objects will be either "characters", objects that move every tick (Bunny, hunters), or "environment", objects that do not move (rewards, traps). Most of the logic will be written in the "character" objects, as they will handle what happens when they run over a trap or a reward. Specifically, the Bunny object will get the current keyboard inputs and act accordingly. Of note is that a wall will be an "environment" object.

The user interface updates are called from the "game" object based on information gathered from the "maze" object. It may also be able to handle additional user inputs such as pausing the game.