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 forest; upon collecting the necessary eggs, a portal will open allowing the user to enter and win the game. The Bunny also has a score, if that value falls below zero the game will be lost.

There will be traps scattered around the forest that have various effects. 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 decrease their score. Inversely, there will also be rewards scattered around the forest that can, for example, add to the score or make the Bunny temporarily invulnerable.

There will also be enemies—hunters, wolves—trying to harm the Bunny. If the Bunny runs into an enemy then the game will be lost. Specifically, the hunter enemies also have the ability to drop traps.

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

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 “environment” objects. This class of objects will contain the walls, rewards, and traps. The “maze” will also contain a list of “character” objects (such as the Bunny or the enemies). Most of the logic will be written in these “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.

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