Assessment idea

**Elevator pitch**

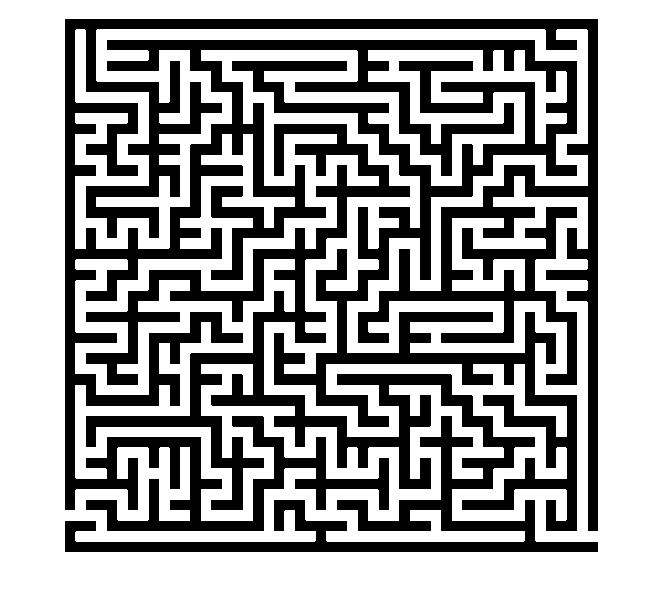
This project will demonstrate how different behaviors interact in a maze, with a townsperson trying to make their way through the maze whilst a knight is going to be wandering around randomly until it is in range then it will follow the townsperson.

**Movement**

Using an A\* library.

I’m going to be utilizing a grid based system to assign node to the paths of my maze to allow the pathfinding algorithm to search through the maze and find a path.

Map



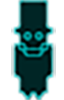
End Point

Start Point

**Agents**

For my project I intend to have two agents in the maze with two separate types of behavior each. The two agents I intend to have will be a trapped townsperson and a knight that is patrolling the maze for intruders.

Townspeople

Townspeople - Behaviors

Trying to find its way through the maze from the start point.

Behaviors

- Flee

The townsperson will flee the knight if the knight enters the townspersons inner radius

- Follow path

This will be the default behavior for the townsperson whilst the townsperson isn’t close enough to the knight. The path that the townsperson will follow will be from their current position to the exit of the maze.

Knights

Knights - Behaviors

Trying to find the townsperson in the maze.

- Follow

The knight will follow the townsperson if they enter the knight’s radius

- Wander

This will be the default behavior for the knight whilst not able to see the townsperson

Agent Radii

To allow the different agents to interact and change which behavior they are currently using.

Agent

Inner Radius

Outer Radius

Rule for path swaps

|  |  |  |
| --- | --- | --- |
| Townspeople | Flee -> Follow path | Follow path -> Flee |
| Rule for swap | Knight exits the townspersons outer radius | Knight enters the townspersons inner radius |

|  |  |  |
| --- | --- | --- |
| Knight | Follow -> Wander | Wander -> Follow |
| Rule | Townsperson exits the knights outer radius | Townsperson enters the knights inner radius |

Post mortem

This project was very long but ultimately rewarding because of the logic I ended up having to think through and problems I had to solve so that the program would work well. Mainly the issues I had centered around the logic for the wandering of entities around the scene and making that last indefinitely. There was this issue where the entities would find a path of zero nodes which made it so that the entity’s logic couldn’t tell that it wasn’t moving anywhere and break from the follow path behavior (wandering implementation) to find another new random path around the map. At the end of my project I decided to pivot away from the townsperson finding a path to an “exit” as this would end up requiring more work than had time for. I would’ve had to implement a reset position in the update loop for if the townsperson made it to the exit so the program would keep running.