**Maze Generator Plan**

Since my game will be revolved around a maze I would need an algorithm, which would randomly generate a maze so that every level in my game is different. so, I created my own algorithm to randomly generate a maze. This algorithm called Maze Generator should generate a random maze every time the program is ran , so that each level of my game will be different.

Objectives for the Maze Generator

* This maze should be randomly generated, so that every level there will be a completely different maze
* In this maze, every square should be accessible
* This maze could be interpreted as a graph so that it can be traversed for the AI

How the algorithm will work?

**Stage 1:**

The user will decide the maze size and a matrix will be created based on the user’s input

Then every square in the matrix will be filled with a random number, either 0,1,2 or 3.

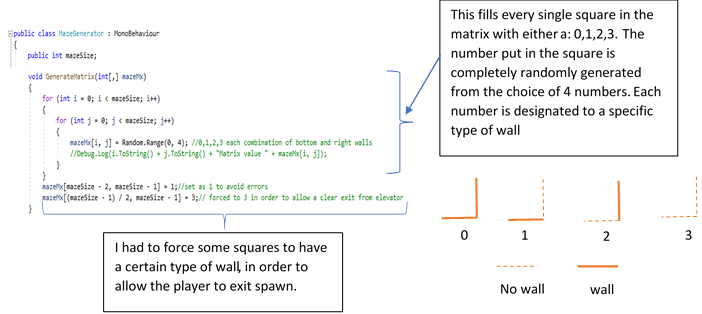
Each number is assigned to a specific type of wall

0 = RIGHT WALL & DOWN WALL

1 = ONLY DOWN WALL

2 = ONLY RIGHT WALL

3 = NO WALL



The numbers in the squares will determine which wall will be placed on that square

**Stage 2:**

After this, a basic maze would’ve been made. However not all the objectives are met. There is a very high possibility that every square is not accessible. So, to fix this there will be a recursive Depth First traversal algorithm. This will check how many squares are accessible from the top left of the maze.

Until the number of squares accessible is not equal to the area of the maze (the users input squared) the algorithm will a break wall and repeat the traversal algorithm until the number of squares accessible is equal to the area.

Flowchart of Maze Generator

START

**Stage 1:**

INPUT MAZE SIZE

MAZESIZE = INPUT

CREATE MATRIX SIZE MAZESIZE\*MAZESIZE

FILL FIRST SQUARE WITH A RANDOM NUMBER RANGING FROM 0-3

COUNT=1

FILL NEXT SQUARE WITH A RANDOM NUMBER RANGING FROM 0-3

END

DISPLAY MAZE

COUNT=COUNT+1

NO

LOAD PREFABS OF WALLS

ASSIGNED TO EACH NUMBER

IS COUNT = MAZESIZE

YES

Example of the algorithm

**Stage 1**

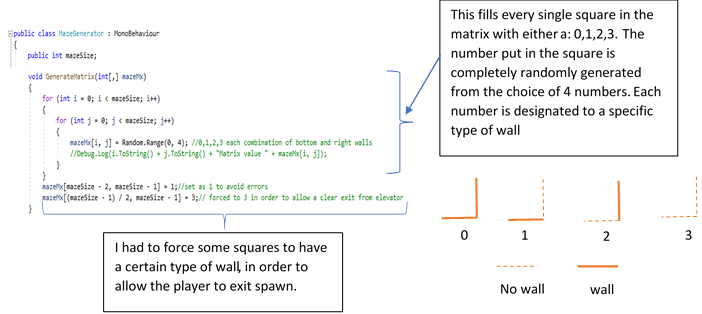
This is an example of how the maze Generator Algorithm works

I decided to set the maze size input as 6, which created a 6\*6 matrix.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

The algorithm then fills every cell in the matrix with a number ranging from 0 to 3, using a random number generator (limiting the numbers generated to either 0,1,2 or 3)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | 0 | 1 | 3 | 0 | 2 |
| 3 | 0 | 2 | 0 | 2 | 2 |
| 2 | 3 | 1 | 0 | 2 | 1 |
| 0 | 1 | 1 | 2 | 3 | 2 |
| 2 | 3 | 3 | 1 | 0 | 0 |
| 3 | 2 | 0 | 1 | 3 | 0 |



WALLS

The algorithm then goes through every cell in the matrix, loading the wall prefab assigned to each specific number. This creates a randomly generated maze

However not all the objectives are met yet, since not every square in the maze is accessible.