**Maze Solver**

Q1.1)

The maze can be seen as a graph where each “-“ path can be seen as a node and each move as an edge. This then creates a simplified graph where the shortest path for the maze is also the shortest path of the graph, so searches used in graphs can be used for the maze as well. By simplifying the maze into a graph certain useless information such as walls, directions can be removed and therefore increase the performance as only the paths which split up are recorded. Therefore, the maze can be easily solved as a graph using any search algorithm on top of that.

Q1.2)1)

The depth-first search algorithm traverses the graph as far down as possible before it backtracks till it reaches a new path and does the same until it reaches the lowest point or goal. This very easily implemented with a stack as u visit the node u add it to the stack, then when you reach a dead end u can pop the last visited node and backtrack.

Q1.2)2)

Path:

( 0 , 1 ) -> ( 1 , 1 ) -> ( 1 , 2 ) -> ( 1 , 3 ) -> ( 1 , 4 ) -> ( 1 , 5 ) -> ( 2 , 5 ) -> ( 3 , 5 ) -> ( 4 , 5 ) ->

( 5 , 5 ) -> ( 5 , 6 ) -> ( 5 , 7 ) -> ( 5 , 8 ) -> ( 6 , 8 ) -> ( 6 , 9 ) -> ( 6 , 10 ) -> ( 6 , 11 ) -> ( 6 , 12 ) -> ( 6 , 13 ) -> ( 6 , 14 ) -> ( 6 , 15 ) -> ( 6 , 16 ) -> ( 6 , 17 ) -> ( 7 , 17 ) -> ( 8 , 17 ) -> ( 8 , 18 ) ->

( 9 , 18 ) -> END

Q1.2)3)

**Maze-Small:**

Number of nodes explored: 82

Time taken: 0.00019288063049316406 seconds

Number of steps in the path: 26

Q1.2)4) You can change the maze file at the top of the code by changing the mazeFile variable to the maze location.

**Maze-Medium:**

Number of nodes explored: 6891

Time taken: 0.016548871994018555 seconds

Number of steps in the path: 508

**Maze-Large:**

Number of nodes explored: 13582

Time taken: 0.03231310844421387 seconds

Number of steps in the path: 1119

**Maze-VLarge:**

Number of nodes explored: 620748

Time taken: 1.328536033630371 seconds

Number of steps in the path: 5724

Q1.3)1)

A different algorithm to solve the maze could be one that use a heuristic approach like the A\* algorithm and best first search. These are very good as they use an estimate of the distance to the goal from each node which means it can find a shorter path. As it has to calculate a heuristic estimate for each node it could mean that its slower even if it looks at less nodes.

Depth-first search has potential to be faster for smaller mazes however A\* should be faster for larger ones as the heuristic guides its searched nodes towards the goal whilst Depth-first search uses a systematic way of exploring nodes using there relative locations and not the goals distance to the nodes which leads in larger mazes to longer times.

Q1.3)2)

Path:

( 0 , 1 ) -> ( 1 , 1 ) -> ( 1 , 2 ) -> ( 1 , 3 ) -> ( 1 , 4 ) -> ( 1 , 5 ) -> ( 2 , 5 ) -> ( 3 , 5 ) -> ( 4 , 5 ) ->

( 5 , 5 ) -> ( 5 , 6 ) -> ( 5 , 7 ) -> ( 5 , 8 ) -> ( 6 , 8 ) -> ( 6 , 9 ) -> ( 6 , 10 ) -> ( 6 , 11 ) -> ( 6 , 12 ) -> ( 6 , 13 ) -> ( 6 , 14 ) -> ( 6 , 15 ) -> ( 6 , 16 ) -> ( 6 , 17 ) -> ( 7 , 17 ) -> ( 8 , 17 ) -> ( 8 , 18 ) ->

( 9 , 18 ) -> END

**Maze-Small:**

Number of nodes explored: 64

Time taken: 0.0002448558807373047 seconds

Number of steps in the path: 26

Q1.3)3)

**Maze-Medium:**

Number of nodes explored: 24085

Time taken: 0.08928775787353516 seconds

Number of steps in the path: 320

**Maze-Large:**

Number of nodes explored: 43740

Time taken: 0.15766620635986328 seconds

Number of steps in the path: 973

**Maze-VLarge:**

Number of nodes explored: 550033

Time taken: 1.1285569667816162 seconds

Number of steps in the path: 3690

Q1.3)4)

My results were very similar to my prediction that for small mazes Depth-first search would be faster and for larger mazes it would be out timed by the A\* search however this was olnly correct for the VLarge maze.

The Large maze was slower which is surprising but its due to the fact A\* searches more than 3 times more nodes before it reaches the goal. This is likely due to the placement of the walls as it would block the search and give it a fake positives whilst its blocked by the heuristic thinking a path is shorter. Meanwhile Depth-first search doesn’t take in mind the goal when selecting the nodes to explore. The VLarge maze and small however were the opposite as A\* explored less nodes probably due to the maze walls locations

My prediction that for every size maze A\* gives a more optimal solution for the number of steps and path was correct for all because of the heuristic nature of the algorithm which is true for all but the small where they were equivalent.