Homework 2

James Fu

Problem 2:

Given the algorithm, main function, and maze shown at the end of problem 1, what are the first 12 (r,c) coordinates popped off the stack by the algorithm?

(5, 6) (6, 6) (4, 6) (4, 7)

(4, 8) (3, 8) (2, 8) (1, 8)

(1, 7) (3, 6), (5, 5), (5,4)

Problem 4:

Given the same main function and maze as are shown at the end of problem 1, what are the first 12 (r,c) coordinates popped from the queue in your queue-based algorithm?

How do the two algorithms differ from each other? (Hint: how and why do they visit cells in the maze in a different order?)

Pop:

(5, 6) (5, 5) (4, 6) (6, 6)

(5, 4) (3, 6) (4, 7) (5, 3)

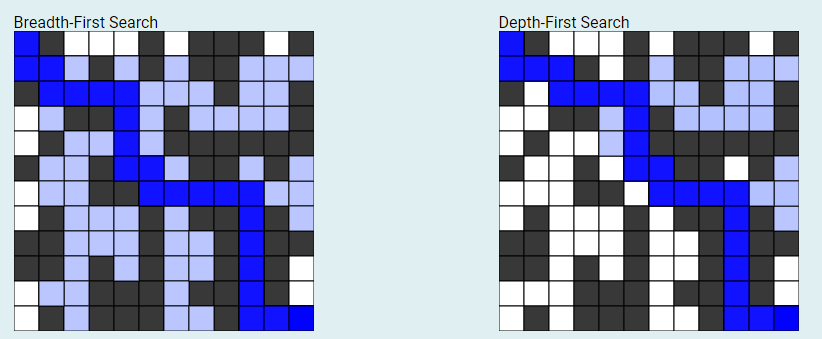
(4, 4) (4, 8) (5, 2) (3, 4)

The two algorithms are different because the first algorithm involves using a stack, while the second algorithm involves using a queue.

A stack algorithm (or depth first) for solving a maze is different because it essentially uses a stack which can access the very last element (e.g. like plates at a buffet). The search goes through a grid as long as there is a valid next element. If it reaches a wall or dead end, the algorithm uses the element right before the last element to determine if a different path along that branch exists.

A queue based algorithm (breadth first search) uses the coordinate added first to test if a maze path is valid, and go to each coordinate following that and determine if there is a path in different directions of that coordinate. After all 4 surrounding coordinates, the algorithm can determine the previously added coordinates next coordinate. Thus, this type of algorithm ensures that the shortest path is found.

Here is a visualization I found on Github (not created by me) since I’m not sure if I’m explaining it very well above.



[https://seanperfecto.github.io/BFS-DFS-Pathfinder](https://seanperfecto.github.io/BFS-DFS-Pathfinder/) (created by user seanperfecto)