3. Design Questions

In a file submitted as DesignQuestions.pdf include the answers to the following questions: 1. Why do the versions (stack vs. queue) of the algorithm output different paths for some of the mazes?

The stack-based algorithm follows a LIFO strategy, which means it explores one direction as far as possible before having to returne to previous points to try an alternate route (backtracking). In contrast, the queue-based algorithm operates in FIFO, systematically expanding all adjacent options at a given stage before proceeding to deeper levels. This difference in exploration order can lead to different paths, the stack-based method may result in a more extended and less direct route, possibly a longer path, and the queue-based method guarantees the shortest when all moves have equal cost.

2. Which implementation (stack vs. queue) explores one path at a time?

The stack-based implementation, exoloring a single route through the maze as far as it can go without interruption, only returning to earlier decision points when it encounters a dead end or a fully explored branch.

3. Which implementation explores all possible paths at the same time?

The queue-based implementation explores all possible paths in parallel by expanding nodes level-by-level. At each stage, it evaluates all reachable adjacent/valid neighbor positions before moving to the next set of positions, a more efficient expansion from the starting point.

4. Which implementation finds the shortest path? Is this guaranteed to be true for all input mazes?

The queue-based implementation always finds the shortest path in any solvable, unweighted maze. The algorithm explores all positions at a given distance from the start before advancing to positions farther away, exploring nodes in order of increasing distance (d+1 nodes after all d nodes visited). This guarantee only applies to unweighted mazes where all movements have equal cost.

5. Give an example input maze where the stack and queue algorithms would find a different path through the maze. This example must be different from all given mazes and examples in this handout.

ex) 4
0 0 1 0
1 0 1 0
0 0 1 0
0 1 0 0