Drew Wan

305751876

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?

The first 12 (r,c) coordinates popped off the stack by the algorithm is

1. (3,4)
2. (3,3)
3. (3,5)
4. (2,5)
5. (1,5)
6. (1,6)
7. (1,7)
8. (1,8)
9. (2,8)
10. (3,6)
11. (4,4)
12. (5,4)

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?)

Here, the first 12 (r,c) coordinates popped from the queue in this algorithm is

1. (3,4)
2. (4,4)
3. (3,5)
4. (3,3)
5. (5,4)
6. (3,6)
7. (2,5)
8. (6,4)
9. (5,5)
10. (1,5)
11. (7,4)
12. (5,6)

The first algorithm uses the top coordinate in the stack or the last added coords to test if the path exists. It will explore one path until it reaches a wall. After, it will use the second to last coordinate to test that path and keep backtracking until it finds a valid path. This can be called a depth-first search.

The second algorithm uses the front coordinate in the queue or the first added coordinate to test the path. It will reach each coord and test if there is a possible path around it. After all 4 are looked at. At the current coordinate, the algorithm will do the neighbor coordinate. This is more systematic because it looks around all the neighbor cells. This can be called breadth-first search.

This means that the second algorithm using queue will most likely find the shortest path, whereas the first algorithm most likely won’t.