## **Problem 2**

**First twelve coordinates visited in mazestack.cpp:**

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

## **Problem 4**

**First twelve coordinates visited in mazequeue.cpp:**

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

The algorithm implemented in *mazestack.cpp*, which uses a stack, visits coordinates with a (pseudo?) depth-first approach. The algorithm implemented in *mazequeue.cpp*, on the other hand, uses a queue and therefore visits coordinates with a breadth-first approach.

In the *mazestack* algorithm, for instance, we push adjacent coordinates (4, 4), (5, 3), and (3, 3) onto the stack while we’re at the starting coordinate (4, 3). We then pop the (3, 3) coordinate (which was added last) and push its (valid) adjacent coordinates, and after visiting each of them, their own adjacent coordinates and so on, we come back to (5, 3) and repeat the same process. We don’t even revisit (4, 4) (the first coordinate pushed to the stack after the starting coordinate) in the entire algorithm, since we find our solution in a path via (5, 3).

In the *mazequeue* algorithm, on the other hand, we start by pushing the same adjacent coordinates (4, 4), (5, 3), (3, 3) onto the stack while we’re at the starting coordinate (4, 3), but we now visit them in the order in which they were added to repeat the algorithm. We therefore visit and pop (4, 4) first (rather than last), followed by (5, 3) and (3, 3). The entire time, we are pushing new coordinates onto the queue (such as the coordinates adjacent to (5, 3) and (3,3)) but we visit and pop them strictly in the order they were added.

We get different sequences of coordinates visited in each approach because of this fundamental difference.