Project #5
CS 2210 – Fall 2021
Sam DeCook

I. Requirements: Restate the problem specification, and any detailed requirements

Use a queue to implement a breadth-first search to find the shortest solution to a given maze. You should start your search from the start position; assume you don't know where the target location is.

II. Design: How did you attack the problem? What choices did you make in your design, and why? Show class diagrams for more complex designs.

First, I implemented a Location and Coord class. Then I broke down the task into parsing, searching, and printing out the solution. Having the problem laid out conceptually helped me to focus on the coding knowing I had the correct conceptual solution.

III. Security Analysis: State the potential security vulnerabilities of your design. How could these vulnerabilities be exploited by an adversary? What would be the impact if the vulnerability is exploited?

These classes don't really give an opportunity for security vulnerabilities to be exploited.

IV. Implementation: Outline any interesting implementation details.

This is more of a quirk. When parsing the maze into the maze array, the coordinates are the loop variable j, i (for x, y). Since the i loop goes line by line and the j loop goes through each line, the j variable is actually the x coordinate (and the i variable the y). I think I also made a small design error in breaking up my if statements in the search loop. However, fixing it would require a total redesign of that section, so I am going to leave it.

V. Testing: Explain how you tested your program, enumerating the tests if possible. Explain why your test set was sufficient to believe that the software is working properly, i.e., what were the range of possibilities of errors that you were testing for.

I did testing through Gradel, which had a good test set. We didn't have to parse for correct input, so that made for limited testing.

VI. Summary/Conclusion: Present your results. Did it work properly? Are there any limitations? If it is an analysis-type project, this section may be significantly longer than for a simple implementation-type project.

It worked properly and I don't believe there are any limitations.

VII. Lessons Learned: List any lessons learned. What might you have done differently if you were going to attack this again?

I might have done the if statements in the search loop differently. Other than that, I'm happy with my work on this project, I didn't make any big mistakes I needed to backtrack from.

My code compiled and ran properly, and produced the expected output