Assignment 2

Hyoungshick Kim

October 8, 2018

1 Finding the shortest path in a maze

This is a programming assignment to test your understanding of Stacks or Queues.

- A *maze* is given as $n \times m$ character matrix of squares where there exist the starting point s and the ending point e.
- Given a maze, your goal is to find a path from s to e on the maze with the minimum length of the path. A path consists of squares which are horizontally or vertically adjacent.
- In the matrix representing a maze, "*" squares are blocked squares while "0" squares are available to explore the maze.
- Interestingly, there are k warp zone locations that allow you an instant travel between those locations. That is, when you move in a warp zone location, you can instantly move to any warp zone without any cost. Surely, you cannot use the warp zone if you don't want it. In our matrix, "w" squares represent warp zone locations.
- You will write a code in the C programming language to find a path from s to e on the maze with the minimum length of the path from a given $n \times m$ matrix to represent a maze in the input file named `hw2_input.txt`'. Please write (1) the length of the path satisfying the above requirements and (2) the path itself, respectively, into the output file named '`hw2_output.txt`'.
- The followings are examples of input and output files:

```
[Input file: hw2_input.txt]
0 w 0 e
0 * 0 *
w s 0 0
0 0 * 0

[Output file: hw2_output.txt]
3
(3,2),(3,1),(1,2),(1,3),(1,4)
```

```
[Input file: hw2_input.txt]
0 0 0 e
0 * * *
0 s 0 w
w 0 * 0

[Output file: hw2_output.txt]
6
(3,2),(3,1),(2,1),(1,1),(1,2),(1,3),(1,4)


[Input file: hw2_input.txt]
0 0 w e
0 * 0 *
0 s * w
w 0 0 0

[Output file: hw2_output.txt]
3
(3,2),(3,1),(4,1),(1,3),(1,4)
```

- If there exist multiple answers with the minimum length of the path, you can only write one of them.
- If there is no answer connecting s with e on a maze, please write "NULL" only.
- You will be judged by (1) the correctness of the results returned by your submitted program, (2) the length of the output path, (3) the actual running time of the program and (4) the well-written document to explain your source code and the performance analysis of your algorithm. For test, we will use $2 \leq n, m \leq 300$ and $2 \leq k \leq 20$. Please test your code extensively with several inputs, so you are sure it works correctly.
- You cannot use any pre-defined data structures except arrays.
- Please upload your source code (c files), instructions to illustrate how your source code works, document to explain your code and the performance analysis to iCampus. Submit your assignment by midnight, Sunday October 21; **late submissions are allowed with a penalty. Each 24 hours (or part thereof) late will cost you 20%.**
- **Your assignments must be your own original work.** We will use a tool to check for plagiarism in assignments.