Task:

Solve a maze.

Requirements:

For this assignment, you are tasked to build a console application that will solve a given maze.

The maze will be given to you as a simple text file. The file will contain a maze that will use the pipe (|), hyphen (‐) and plus sign (+) to represent the maze walls.

Example:

Maze Start

Location

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| | | | | | |

+-+-+-+-+-+ +-+ + +-+-+ + +-+ + +-+ + + +-+-+ + + +

| | | | | | | | | | | | | | |

+ + + +-+-+ + +-+ + +-+-+ + +-+ + + + +-+-+ +-+-+ +

| | | | | | | | | | | | | | | |

+ +-+ + +-+-+-+ + + +-+ + + + +-+-+-+-+-+ +-+ +-+-+

| | | | | | | | | | | | | | |

+ + +-+ + +-+ + + +-+ + + + + + + +-+-+-+-+ +-+-+ +

| | | | | | | | | | | | | | | |

+ +-+ +-+ + +-+ +-+ +-+ + +-+-+ + + +-+ +-+ +-+-+-+

| | | | | | | | | | | | |

+-+ +-+-+-+-+ +-+-+ +-+ + + +-+-+ +-+-+-+ + + + + +

| | | | | | | | | | | | |

+ +-+-+ +-+ + + + +-+ +-+-+-+ + +-+ + + +-+-+ + + +

| | | | | | | | | | | | | |

+ + + +-+ + + +-+ + +-+-+-+-+-+ + + + +-+-+-+-+-+-+

| | | | | | | | | | | | | | |

+ + + + +-+ + + +-+ +-+ + +-+ + + +-+-+-+-+ + +-+ +

| | | | | | | | | | | | | |

+-+-+ +-+ +-+-+-+ +-+ +-+-+-+ +-+-+ +-+ + +-+ + +-+

| | | | | | | | | | | | | |

+ + +-+-+ + +-+ +-+ +-+-+-+ + +-+ +-+-+ +-+ + +-+ +

| | | | | | | | | | | |

+ +-+ + +-+ + +-+ +-+ +-+-+-+ + + + +-+-+-+-+ + +-+

| | | | | | | | | | | | | |

+ + +-+ + + +-+-+-+ +-+-+ + +-+-+-+ +-+ + +-+-+-+ +

| | | | | | | | | | | | | |

+ +-+-+-+-+-+ + +-+ + + + +-+ +-+ + +-+-+-+-+ + + +

| | | | | | | | | | | | | | |

+ + +-+ + + + + + +-+-+ + + +-+ +-+-+-+-+ +-+-+ +-+

| | | | | | | | | | | | | |

+-+-+ + + + + + + +-+ + +-+-+ +-+ +-+-+-+-+ + +-+-+

| | | | | | | | | | | | | | | | | |

+ + + + + + + +-+ + +-+ + + + + +-+ + + + + + + + +

| | | | | | | | | | | | | | | | | | |

+ +-+ + +-+ +-+ +-+-+ + +-+-+-+ + +-+ +-+-+ +-+-+ +

| | | | | | | | | | | | | | |

+ + + +-+ + + + + + + +-+ + + +-+-+ +-+-+ +-+ +-+-+

| | | | | | | | | | | | |

+-+-+-+ +-+-+ +-+ +-+ +-+-+ + + +-+-+-+-+ + + + + +

| | | | | | | | | | | |

+ +-+ +-+-+-+-+ +-+ +-+-+-+-+ + +-+ + +-+-+-+-+ + +

| | | | | | | | | | |

+-+ +-+ + +-+-+-+ +-+ +-+-+ + + + +-+-+-+-+-+ +-+ +

| | | | | | | | | | | | |

+ + + +-+ + +-+-+-+ +-+ + +-+-+-+-+ + +-+ + +-+-+ +

| | | | | | | | | | | | | | | |

+ +-+-+-+-+ + +-+-+ + + + +-+ + + + +-+ +-+-+ + + +

| | | | | | | | |

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Maze Exit

Your application will then read the maze file and, using stacks and/or queues, produce the solution to the maze as a second text file. The solution through the maze will be noted using another character, such as the pound sign (#).

The maze solution file will resemble the following:

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

############| | | | | | |

+-+-+-+-+-+#+-+ + +-+-+ + +-+ + +-+ + + +-+-+ + + +

|#####| #| | | | | | | | | | | | |

+#+ +#+-+-+#+ +-+ + +-+-+ + +-+ + + + +-+-+ +-+-+ +

|#| |#|#####| | | | | | | | | | | |

+#+-+#+#+-+-+-+ + + +-+ + + + +-+-+-+-+-+ +-+ +-+-+

|#|###|###| | | | | | | | | | | |

+#+#+-+ +#+-+ + + +-+ + + + + + + +-+-+-+-+ +-+-+ +

|#|###| |#| | | | | | | | | | | |

+#+-+#+-+#+ +-+ +-+ +-+ + +-+-+ + + +-+ +-+ +-+-+-+

|###|#####| | | | | | | | | | |

+-+#+-+-+-+-+ +-+-+ +-+ + + +-+-+ +-+-+-+ + + + + +

| ##### | | | | | | | | | | | |

+ +-+-+#+-+ + + + +-+ +-+-+-+ + +-+ + + +-+-+ + + +

| | | ###| | | | | | | | | | |

+ + + +-+#+ + +-+ + +-+-+-+-+-+ + + + +-+-+-+-+-+-+

| | | |###| | | | | | | | | | |

+ + + +#+-+ + + +-+ +-+ + +-+ + + +-+-+-+-+ + +-+ +

| | |###| | | | | | | | | | |

+-+-+ +-+#+-+-+-+ +-+ +-+-+-+ +-+-+ +-+ + +-+ + +-+

|###| |#| | | | | | | | | | |

+#+#+-+-+#+ +-+ +-+ +-+-+-+ + +-+ +-+-+ +-+ + +-+ +

|#|###|###| | | | | | | | |

+#+-+#+#+-+ + +-+ +-+ +-+-+-+ + + + +-+-+-+-+ + +-+

|#|###|#| | | | | | | | | | |

+#+#+-+#+ + +-+-+-+ +-+-+ + +-+-+-+ +-+ + +-+-+-+ +

|#|#####| |### | | | | | | | |###| |

+#+-+-+-+-+-+#+#+-+ + + + +-+ +-+ + +-+-+-+-+#+#+ +

|#|#####| |#|#| | | | | | | |#####|# |

+#+#+-+#+ + +#+#+ +-+-+ + + +-+ +-+-+-+-+#+-+-+#+-+

|###|###| | |#|#|#####| | | |#########| |### |

+-+-+#+ + + +#+#+#+-+#+ +-+-+ +-+#+-+-+-+-+ +#+-+-+

| |#| | | |#|###|###| | | |###| | | |#|###|

+ + +#+ + + +#+-+ +#+-+ + + + +#+-+ + + + + +#+#+#+

| | |#| | | |###| |###| | | |#| | | |###|#|

+ +-+#+ +-+ +-+#+-+-+#+ +-+-+-+#+ +-+ +-+-+ +-+-+#+

| | #| | | |#| |#| |###| | | |#####|

+ + +#+-+ + + +#+ + +#+-+ + +#+-+-+ +-+-+ +-+#+-+-+

| |###| |###| | # | |#| | |###| |

+-+-+-+#+-+-+#+-+ +-+#+-+-+ +#+ +-+-+-+-+ + + +#+ +

| |#######| |###| |#| | | |#| |

+ +-+ +-+-+-+-+ +-+#+-+-+-+-+#+ +-+ + +-+-+-+-+#+ +

| | | |###|#######|#| | | |###|

+-+ +-+ + +-+-+-+#+-+#+-+-+#+#+ + +-+-+-+-+-+ +-+#+

| | | |#######|###| |###| | | | | #|

+ + + +-+ +#+-+-+-+#+-+ + +-+-+-+-+ + +-+ + +-+-+#+

| | | |#|#######| | | | | | | | | |#|

+ +-+-+-+-+#+#+-+-+ + + + +-+ + + + +-+ +-+-+ + +#+

| |###| | | | | | | ##

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

**Notes**:

You may assume the maze is always a grid. However, you cannot assume the maze itself is always the same (i.e. the solution could change, the dimensions may change as well). To achieve full marks your program must be able to solve all of the test mazes provided. These vary in size (length and width), solution path, and line endings (Windows vs. Unix).

You can also assume that the maze always starts at the top left‐hand corner and ends at the bottom right‐hand corner.

Evaluation:

This assignment is worth 36 marks. Please see the marking rubric below.

Assignment Notes:

The assignment must be demonstrated to the instructor on or before the due date during class.

If your assignment is late please send an e-mail to the instructor, hal.o’connell@nscc.ca, to confirm submission. This e-mail will constitute the timestamp for evaluating any late penalty the assignment may incur.

**Marking Rubric**

**Name(s):**

| Criteria | Below Standard | Developing | Acceptable | | Professional | Marks |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | | 3 |
| Stack/Queue NOTE Multiplier  Max of 6 marks here! | * Did not use a stack or queue to solve the maze | * a stack or a queue was used, but it was not used to solve the maze * recursion was used * many small or a few large errors exist | * a stack and/or a queue was used to solve the maze * recursion was not used * a few small errors exist | | * a custom built stack and/or a queue was designed and used to solve the maze * no recursion was used * no errors in implementing the stack or queue | \_x2  \_\_\_\_ |
| Maze PER MAZE!  Max of 15 marks here! | * Did not find the exit of the maze | * the maze exit was found and a path from the entrance to the exit exists * a single path was not found * includes paths to dead-ends * many small or a few large errors exist | * a single path to the * exit was found * no dead-ends are shown * a few small errors in the solution or the display exist | | * the single solution to from the entrance to the exit of the maze was found and displayed using a special character. * the solution file for the example maze, generated by the program, matches the example solution | Maze  1  2  3  4  5  Total  \_\_\_\_\_ |
| Aesthetics | * incorrect or non existent use of whitespace in output * output is confusing and hard to follow | * fair use of   whitespace   * most output is clear, but poorly presented | * good use of whitespace * output is clear and fairly well presented | | * excellent use of whitespace * output is clear and attractively presented | \_\_\_\_\_ |
| Readability | * source code is poorly organized and very difficult to read | * source code is fairly easy to read, but is hard to follow in some areas | * source code is exceptionally well organized and easy to follow | |  | \_\_\_\_\_ |
| Reusability | * source code cannot be reused * no functions or classes used | * portions of code could be reused with modifications | * source code could be easily reused with few modifications | |  | \_\_\_\_\_ |
| Efficiency | * contains large portions that could have been easily reduced using a different method * too much code is replicated, copy /pasted | * tried some methods to improve efficiency * can explain what they attempted | * very clean and efficient code * can propose new ideas for improvement | |  | \_\_\_\_\_ |
| Comments | * little to no comments used | * comments are used, some are meaningful and easily understood * some files and functions have headers | * not over/under commented * comments are meaningful and easily understood * files and functions have headers * Code is self-documenting | |  | \_\_\_\_\_ |
| **Naming** Convention | * no standard naming convention followed | * a standard naming convention was used for part of the program, but deviated often | * industry standard naming convention used throughout the program | |  | \_\_\_\_\_ |
| Consistency | * no consistency in formatting or layout of source code | * source code formatting was present but inconsistent with whitespace, brackets, etc | * source code formatting never deviated from the programmer’s layout | |  | \_\_\_\_\_ |
| Total | | **36** |

0 - Assignment not submitted or work not original.