PROGRAMMING ASSIGNMENT 4

Submission Date : 20.12.2019
Due Date : 03.01.2020 23:59
TA : Necva BÖLÜCÜ

Accept your *4th Assignment*.

Introduction:

In this experiment, you will get familiar with file operations, lists and dictionaries and recursion while you implement program solving mazes. The mazes will be provided with simple text files and the programs output will be the correct path through the maze.

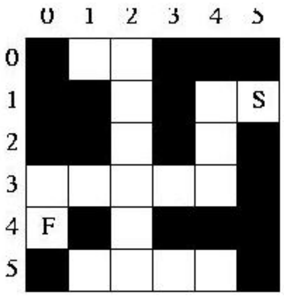
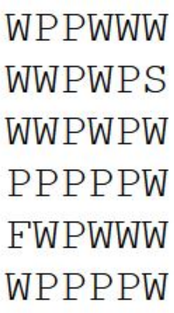


Figure 1: A sample maze with its rows and columns



```
WPPWWW
WWPWPS
WWPWPW
PPPPPW
FWPWWW
WPPPPW
```

Figure 2: The maze pictured above is described by the text file

Assignment:

Imagine it's the first day of BBM103 and you're trying to find your way around the computer engineering building. You're trying to make it to lecture on time. But you're lost! Corridors double back on themselves. You decide to follow the corridor and it somehow leads you to the floor above, though you have no memory of climbing stairs. You're running out of food and water.

In this totally plausible scenario, what do you do? You could strike out in random directions, bouncing off walls until you're lucky enough to get out and see your family again. Or you could be strategic. Pick a path, follow it until it either doubles back or reaches your destination. Do this with every possible path and, maybe by the end of the semester, you'll know a path to your BBM103 lecture! Alternatively, you could use a computer to solve the maze for you, and have it show you the path to take. For simplicity, the maze cannot have loops of any kind.

Solving Maze

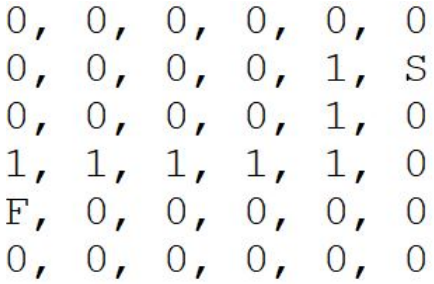
The aim of this assignment is solving the problem of finding and marking a solution path using recursion. The program will find a path by walking the maze recursively by visiting each cell and avoiding walls and already visited cells.

First, you will read input file to construct your maze. Your algorithm must be **recursive**. For this reason, we need to view the problem in terms of similar subproblems. In this case, that means we need to "find a path" in terms of "finding paths."

Remember that a recursive algorithm has at least 2 parts:

- Base case(s) that determine when to stop.
- Recursive part(s) that call the same algorithm (i.e., itself) to assist in solving the problem.

After finding a path for the maze, you will write the path to the output file.



```
0, 0, 0, 0, 0, 0
0, 0, 0, 0, 1, S
0, 0, 0, 0, 1, 0
1, 1, 1, 1, 1, 0
F, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0
```

Figure 3: A solution for the given sample maze

Solving Maze with Health Condition

In this section, you have a finite health. it will be given from command line. With given health, finishing the maze is impossible. In the maze, there is a health bonus. To finish the maze, you should take this health bonus. For this reason, you will find a solution recursively by taking the health bonus.

You will solve the maze without health condition and with health condition and print the solutions to output files.

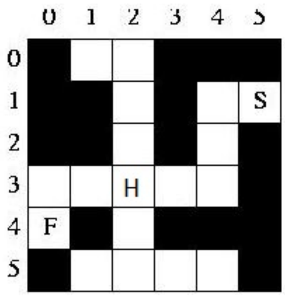


Figure 4: A sample maze with its rows and columns

Grading

- Reading Files: 5
- Solving Maze 35
- Solving Maze with health condition 55
- Writing files: 5

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

Important Notes

- Do not miss the submission deadline.
- Compile your code on dev.cs.hacettepe.edu.tr before submitting your work to make sure it compiles without any problems on our server.
- Save all your work until the assignment is graded.
- The assignment must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating. You can ask your questions via Piazza and you are supposed to be aware of everything discussed on Piazza. You cannot share algorithms or source code. All work must be individual! Assignments will be checked for similarity, and there will be serious consequences if plagiarism is detected.
- You may assume that the input files will be given as command-line arguments in the following order:

```
python3 assignment4.py maze.txt maze_health.txt health_time output.txt
```