AM5801: Computational Lab

Assignment 3

Date: August 28, 2025

Deadline: September 3, 2025 Max mark: 50

1. A maze (table with collection of paths, typically from an source to a destination) is represented as a NxM binary matrix of cells where is each cell can be uniquely identified as (i, j). Consider a rat is initially positioned at cell (0, 0) i.e.. maze[0][0] and the rat wishes to eat food which is present at cell (N-1, M-1) in the maze. However, in the maze, only some of the cells are accessible; other cells have obstacles, hence cannot be accessed. The obstacle and accessible information is stored in the maze matrix where maze[i][j]=1 indicates that the cell (i,j) has an obstacle and maze[i][j]=0 indicates that the cell (i, j) can be accessed to reach the destination. The rat can move in four directions- up, down, left, and right (not diagonally) from one cell to next cell (provided that cell does not have any obstacle). Consider the example given below. The maze is a 5x5 matrix with a value of 1 representing that the cell has an obstacle and a value of 0 indicating that the cell is accessible and can be used as a passage by the rat. The path marked with the grey color indicates the path taken by the rat to go from the source (0,0) to destination (4,4).

0	1	0	1	1
0	0	0	0	0
1	0	1	0	1
0	0	1	0	0
1	0	0	1	0

   (a) Write a program in python which decides if the rat, initially positioned at cell (0, 0), can reach the food, located at cell (N-1, M-1) by traversing only the accessible cells. Essentially, the task is to check if there exists any path from source cell (0,0) to destination cell (N-1, M-1) and print the path if it exists, otherwise print "No solution". This problem can be solved by a recursive function, which implements the following approach. First check if the current cell is the destination. If yes, visit the cell and return 1. Otherwise, check if the rat is currently inside the maze, the current cell does not have any obstacle, and this cell is previously not visited. If true, then try to move in the four direction recursively. If successful, return 1. Finally, if the rat is unable to move in any of the directions, return 0. Complete the following program. **(20)**

(b) Make your solution general by using the variables such as N,M whenever necessary, instead of writing the answer for a particular case. Your solution should work even if we change the value of SIZE to 10. **(10)**

(c) What is the time complexity of your approach in the worst case and why? **(10)**

(d) If diagonal movement were allowed in the maze, what exact changes would you need to make to your function? **(10)**