DS2030 Data Structures and Algorithms for Data Science
Lab 2 (Take Home) Due on August 19, 11.59pm

Instructions

- You are to use Python as the programming language. Use may use Visual Studio Code (or any other editor you are comfortable with) as the IDE.
- You have to work individually for this lab.
- You are not allowed to share code with your classmates nor allowed to use code from the internet. You are encouraged engage in high level discussions with your classmates; however ensure to include their names in the report/code documentation. If you refer to any source on the Internet, include the corresponding citation in the report/code documentation. If we find that you have copied code from your classmate or from the Internet, you will get a straight fail grade in the course.
- The submission must be a zip file with the following naming convention - rollnumber.zip. The Python files should be contained in a folder named after the question number.
- Include appropriate comments to document the code. Include a `read me` file containing the instructions on for executing the code. The code should run on institute linux machines.
- Upload your submission to moodle by the due date and time. Do not email the submission to the instructor or the TA.

This lab will improve your understanding of recursion and stacks.

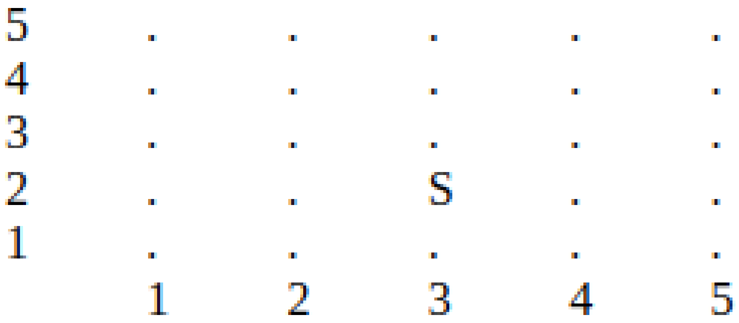
1 Problem Statement: Charging Mr Safayi (10 points)

The objective of this problem is to help Mr Safayi, the cleaning robot to reach its charging station with the valid movement. Design a recursive procedure to make Safayi reach the charging station.
Mr Safayi(S) cleans different places in a house. You have to find the total number of paths Mr Safayi can take to reach the charging station. Solve the problem for following two cases.

Case 1: With Fixed Charging Station
The charging station is fixed at (1,1). The possible moves Mr Safayi can take from any other location to move towards the charging station are West/South (i.e., Left/Down).
To understand, suppose that Mr Safayi (marked as 'S' in the figure) is now working/cleaning the point (2, 3) as shown in the Figure 1, and it suddenly senses that the charge is critically low and needs to charge itself for continuing its cleaning tasks. The charging station is at the location (1, 1). For example, in this Figure 1 there are three possible routes as follows:

- Move left, then left, then down.
- Move left, then down, then left.
- Move down, then left, then left

Implement a recursive function that returns the number of paths Mr Safayi could take back to the origin from the specified starting position, subject to the condition that Safayi can only move west or south (left or down in the diagram).



```
5   .   .   .   .   .
4   .   .   .   .   .
3   .   .   .   .   .
2   .   .   S   .   .
1   .   .   .   .   .
    1   2   3   4   5
```

Figure 1: A sample house that takes a rectangular grid shape

Case 2: Fixed, but Random Charging Station

The charging station can be at any location. However, Mr Safayi is allowed to take only two of the possible four movements East/West/South/North (i.e., Right/Left/Down/Up) to reach the charging station. First figure out how to automate the process of selecting the movements for a given configuration (charging station location). Then find the number of paths to reach the charging station, by modifying the recursion function defined for Case 1 using only the two previously discovered movements.

2 Input Format

1. Input 1: N integer, dimension of the square grid of the house.
2. Input 2: x y integer, Position of Mr Safayi in the grid
3. Input 3: c_x c_y integers, the charging station location; default is 1 1 for case 1.

   The input should be provided as arguments to your Python code. For example mypath.py 5 2 3 1 1;

3 Output Format

1. Total number of paths that can be taken by Mr Safayi. For example: 3 for the problem described in the previous section.

4 References

1. M Goodrich, R Tamassia, and M. Goldwasser, “Data Structures and Algorithms in Python”, 1st edition, Wiley, 2013.