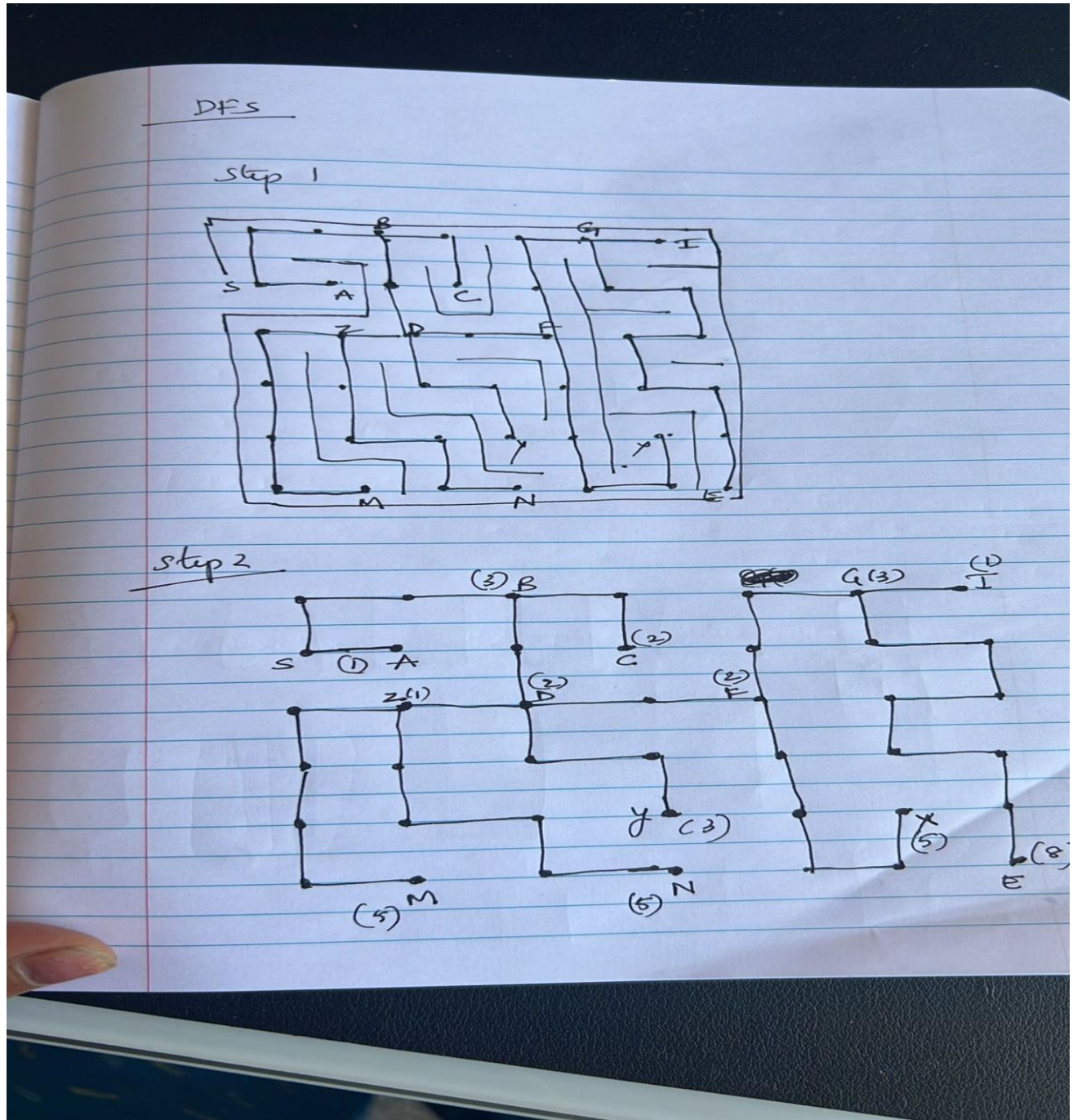


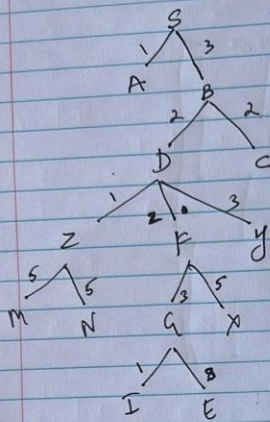
Week 11: Homework 1: Depth-First Traversal: The Maze

Q3 => Conduct Depth First Traversal (DFT) on a maze



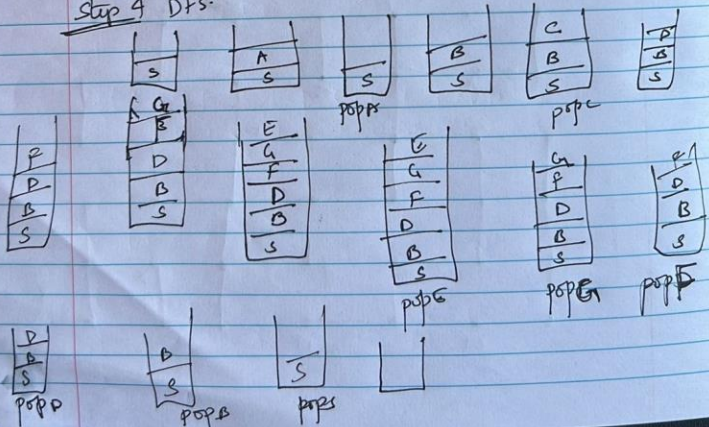
Week 11: Homework 1: Depth-First Traversal: The Maze

Step 3
Tree.



Step 4 DFS.

right, left, up, down



Week 11: Homework 1: Depth-First Traversal: The Maze

Code

```
from typing import List

class Solution:
    def hasPath(self, maze: List[List[int]], start: List[int], destination: List[int]) -> bool:
        m = len(maze)
        n = len(maze[0])
        dirs = [0, 1, 0, -1, 0]

        seen = set()

        def isValid(x: int, y: int) -> bool:
            return 0 <= x < m and 0 <= y < n and maze[x][y] == 0

        def dfs(i: int, j: int) -> bool:
            if [i, j] == destination:
                return True
            if (i, j) in seen:
                return False

            seen.add((i, j))

            for k in range(4):
                x = i
                y = j
                while isValid(x + dirs[k], y + dirs[k + 1]):
                    x += dirs[k]
                    y += dirs[k + 1]
                if dfs(x, y):
                    return True

            return False

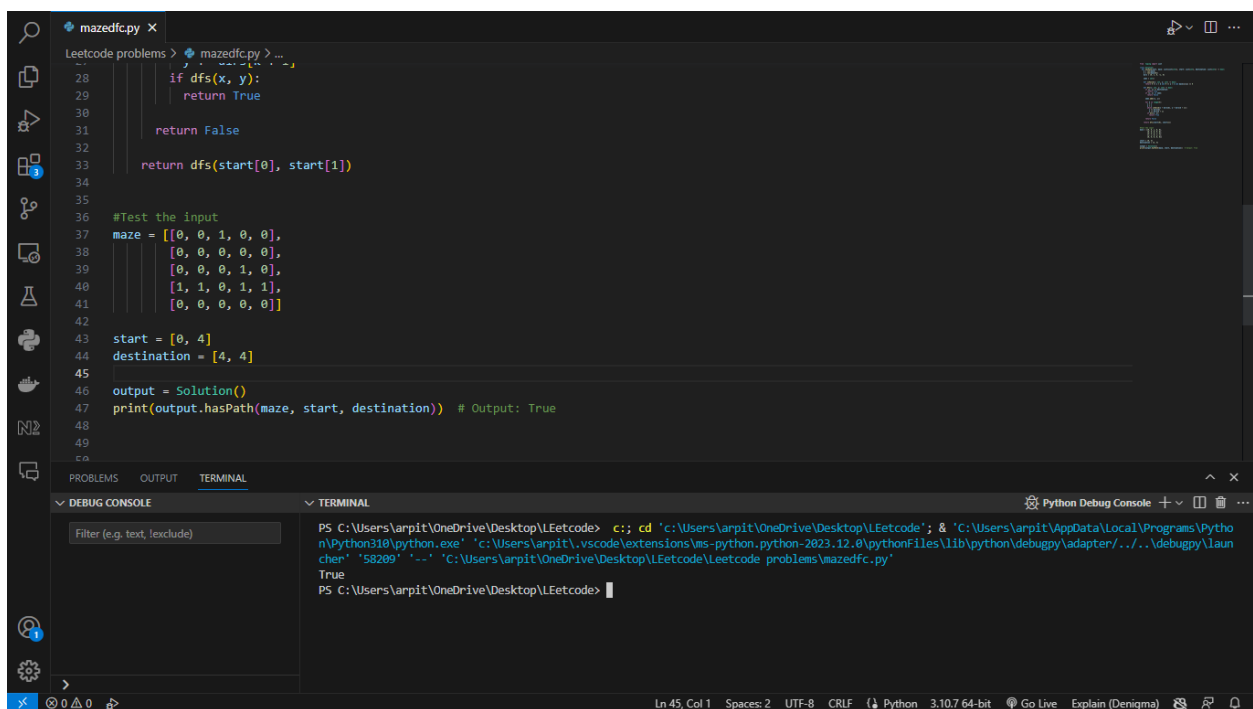
        return dfs(start[0], start[1])
```

Result with different test scenarios :

Week 11: Homework 1: Depth-First Traversal: The Maze

```
maze = [[0, 0, 1, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 1, 0],
        [1, 1, 0, 1, 1],
        [0, 0, 0, 0, 0]]
```

```
start = [0, 4]
destination = [4, 4]
```



```
mazedfc.py
Leetcode problems > mazedfc.py > ...
28     if dfs(x, y):
29         return True
30
31     return False
32
33     return dfs(start[0], start[1])
34
35
36 #Test the input
37 maze = [[0, 0, 1, 0, 0],
38         [0, 0, 0, 0, 0],
39         [0, 0, 0, 1, 0],
40         [1, 1, 0, 1, 1],
41         [0, 0, 0, 0, 0]]
42
43 start = [0, 4]
44 destination = [4, 4]
45
46 output = Solution()
47 print(output.hasPath(maze, start, destination)) # Output: True
48
49
50
```

PROBLEMS OUTPUT TERMINAL

DEBUG CONSOLE

Filter (e.g. text, exclude)

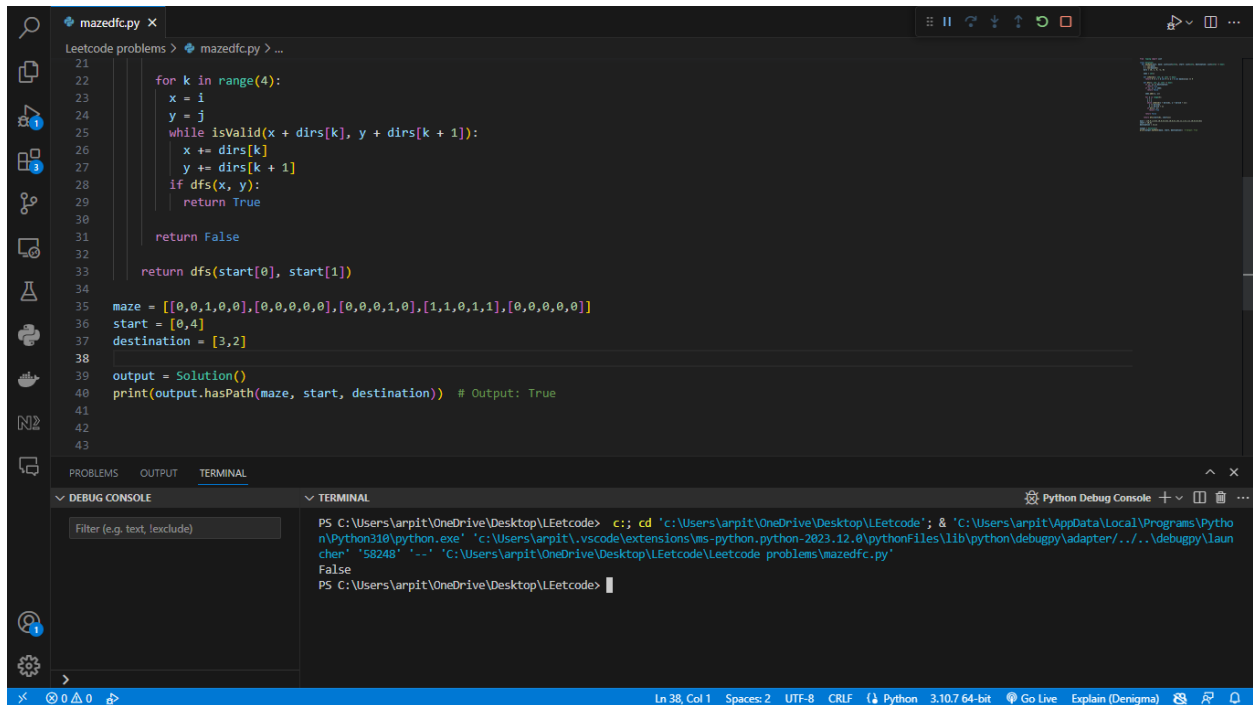
TERMINAL

```
PS C:\Users\arpit\OneDrive\Desktop\Leetcode> c:\cd 'c:\Users\arpit\OneDrive\Desktop\Leetcode'; & 'C:\Users\arpit\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\arpit\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '58269' '-.-' 'C:\Users\arpit\OneDrive\Desktop\Leetcode\Leetcode problems\mazedfc.py'
True
PS C:\Users\arpit\OneDrive\Desktop\Leetcode>
```

Ln 45, Col 1 Spaces: 2 UTF-8 CRLF Python 3.10.7 64-bit Go Live Explain (Denigma)

```
maze = [[0,0,1,0,0],[0,0,0,0,0],[0,0,0,1,0],[1,1,0,1,1],[0,0,0,0,0]]
start = [0,4]
destination = [3,2]
```

Week 11: Homework 1: Depth-First Traversal: The Maze



```
21
22 for k in range(4):
23     x = i
24     y = j
25     while isValid(x + dirs[k], y + dirs[k + 1]):
26         x += dirs[k]
27         y += dirs[k + 1]
28         if dfs(x, y):
29             return True
30
31     return False
32
33 return dfs(start[0], start[1])
34
35 maze = [[0,0,1,0,0],[0,0,0,0,0],[0,0,0,1,0],[1,1,0,1,1],[0,0,0,0,0]]
36 start = [0,4]
37 destination = [3,2]
38
39 output = Solution()
40 print(output.hasPath(maze, start, destination)) # Output: True
41
42
43
```

DEBUG CONSOLE

Filter (e.g. text, exclude)

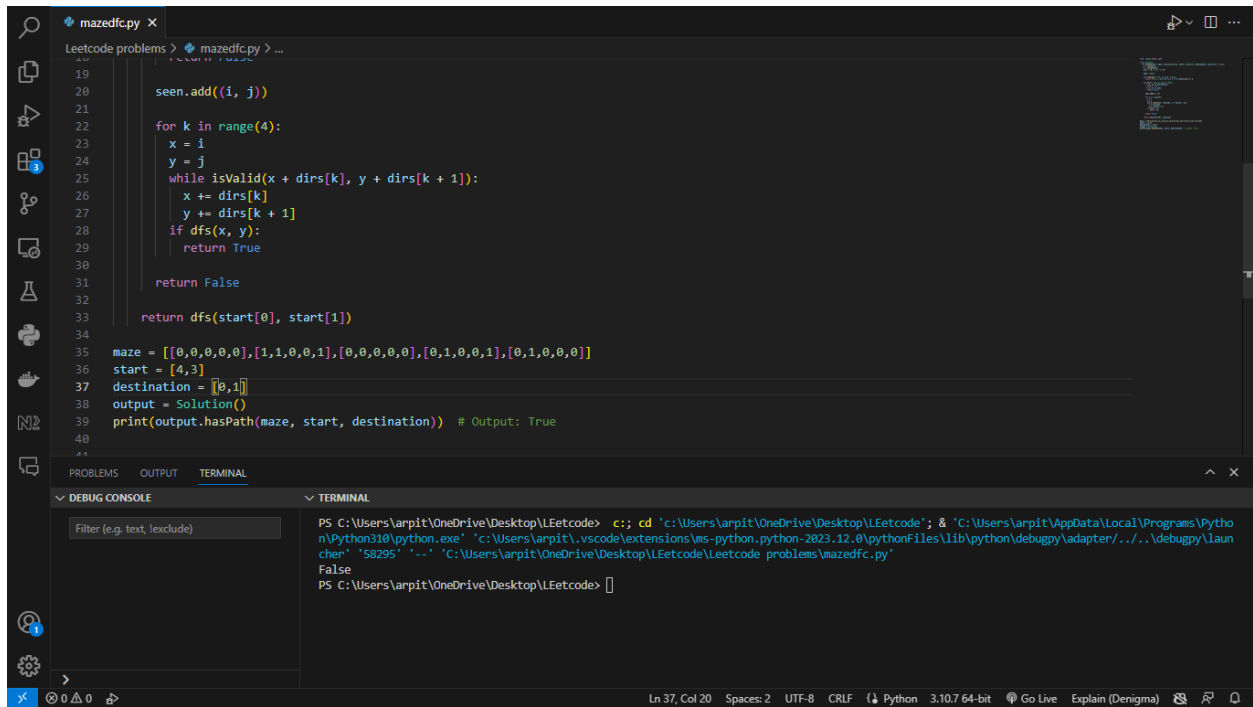
TERMINAL

```
PS C:\Users\arpit\OneDrive\Desktop\LEetcode> cd 'c:\Users\arpit\OneDrive\Desktop\LEetcode'; & 'C:\Users\arpit\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\arpit\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '58248' '--' 'C:\Users\arpit\OneDrive\Desktop\LEetcode\Leetcode problems\mazedfc.py'
False
PS C:\Users\arpit\OneDrive\Desktop\LEetcode>
```

Ln 38, Col 1 Spaces: 2 UTF-8 CRLF Python 3.10.7 64-bit Go Live Explain (Denigma)

```
maze = [[0,0,0,0,0],[1,1,0,0,1],[0,0,0,0,0],[0,1,0,0,1],[0,1,0,0,0]]
start = [4,3]
destination = [0,1]
```

Week 11: Homework 1: Depth-First Traversal: The Maze



```
19
20
21     seen.add((i, j))
22
23     for k in range(4):
24         x = i
25         y = j
26         while isValid(x + dirs[k], y + dirs[k + 1]):
27             x += dirs[k]
28             y += dirs[k + 1]
29             if dfs(x, y):
30                 return True
31
32     return False
33
34     return dfs(start[0], start[1])
35
36 maze = [[0,0,0,0,0],[1,1,0,0,1],[0,0,0,0,0],[0,1,0,0,1],[0,1,0,0,0]]
37 start = [4,3]
38 destination = [[0,1]]
39 output = Solution()
40 print(output.hasPath(maze, start, destination)) # Output: True
```

DEBUG CONSOLE

TERMINAL

```
PS C:\Users\arpit\OneDrive\Desktop\LEetcode> c:: cd 'c:\Users\arpit\OneDrive\Desktop\LEetcode'; & 'C:\Users\arpit\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\arpit\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '58295' '-.' 'C:\Users\arpit\OneDrive\Desktop\LEetcode\Leetcode problems\mazedfc.py'
False
PS C:\Users\arpit\OneDrive\Desktop\LEetcode>
```

GitHub URL: https://github.com/jesalshah14/Maze_DFS_Project