Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year:2025), B.Sc. in CSE (Day)

Lab Performance 04: Graph Traversal

Course Title: Artificial Intelligence Lab

Course Code: CSE-316 Section:221-14

Student Details

Name		ID
1.	Md.Mohibullah	221902083

Submission Date : 26-02-2025

Course Teacher's Name : Md. Sabbir Hosen Mamun

Lab Report Status

Marks: …… Signature:.....

Comments:..... Date:.....

problem:You are given a 2D grid of integers and a target value. Your task is to implement a Python program using Depth-First Search (DFS) that determines whether there exists a path in the grid such that the sum of the integers along the path equals the target value. The sum is obtained by adding the value of each cell visited in sequence. Movement is allowed only horizontally and vertically, and each cell may be used only once in the construction of the sum.

Case#1Input:
3 3
2 3 4
1 5 2
6 2 1
Target: 11

Case#1Output:
Path found
DFS Traversal Order: [(0,1), (1,1), (1,2), (2,2)]

Case#2Input:
3 3
1 1 1
1 1 1
1 1 1
Target: 10

Case#2Output:
Path not found

Code:

```
class Node:
   def __init__(self, grid, target):
       self.grid = grid
       self.target = target
       self.rows = len(grid)
       self.cols = len(grid[0])
       self.directions = [(0, 1), (1, 0), (0, -1), (-1, 0)]  # Right,
Down, Left, Up
       self.visited = set()
       self.traversal_order = []

   def is_valid(self, x, y, index):
       return (
           0 <= x < self.rows and
```

```
            0 <= y < self.cols and
            (x, y) not in self.visited and
            self.grid[x][y] == self.target[index]
        )

    def dfs(self, x, y, index):
        if index == len(self.target):
            return True

        if not self.is_valid(x, y, index):
            return False

        self.visited.add((x, y))
        self.traversal_order.append((x, y))

        for dx, dy in self.directions:
            if self.dfs(x + dx, y + dy, index + 1):
                return True

        self.visited.remove((x, y))
        self.traversal_order.pop()
        return False

    def target_number(self):
        for i in range(self.rows):
            for j in range(self.cols):
                if self.grid[i][j] == self.target[0]:
                    if self.dfs(i, j, 0):
                        print("Target found")
                        print("DFS Traversal Order:", self.traversal_order)
                        return
        print("target not found")


# Input parsing
def main():
    rows, cols = map(int, input().split())
    grid = [input().split() for _ in range(rows)]
    target = input().strip()
```

```
    target_search = Node(grid, target)
    target_search.target_number()


if __name__ == "__main__":
    main()
```

Output: