1.14 UNIQUE PATHS IN A GRID
(ROBOT MOVEMENT PROBLEM)

Question:

A robot is located at the top-left corner of a m×n grid .The robot can only move either down or right at any point in time. The robot is trying to reach the bottom-right corner of the grid. How many possible unique paths are there?

AIM:

To calculate the number of unique paths for a robot moving from the top-left corner to the bottom-right corner of an m × n grid, moving only down or right.

ALGORITHM:

1. Create a dp[m][n] table.
2. Initialize first row & column with 1.
3. Fill the table using relation dp[i][j] = dp[i-1][j] + dp[i][j-1].
4. Answer = dp[m-1][n-1].

PROGRAM:

```
def unique_paths(m, n):
    dp = [[1]*n for _ in range(m)]
    for i in range(1, m):
        for j in range(1, n):
            dp[i][j] = dp[i-1][j] + dp[i][j-1]
    return dp[-1][-1]

def run_unique_paths():
    m = int(input("Enter number of rows (m): "))
    n = int(input("Enter number of columns (n): "))
    print("Number of unique paths:", unique_paths(m, n))

run_unique_paths()
```

Input:

m = 7, n = 3

Output:

```
Enter number of rows (m): 7
Enter number of columns (n): 3
Number of unique paths: 28
>>>
```

RESULT:

Thus, the program is successfully executed, and the output is verified.

PERFORMANCE ANALYSIS:

- Time Complexity: O(m * n)
- Space Complexity: O(m * n) (can be reduced to O(n) with 1D DP).