**75 Days of Code**
**Day 63**
**Problem no : Leetcode 62**
**Problem Title :Unique Paths**
**Problem type : DP**

**There is a robot on an m x n grid. The robot is initially located at the top-left corner (i.e., grid[0][0]). The robot tries to move to the bottom-right corner (i.e., grid[m - 1][n - 1]). The robot can only move either down or right at any point in time.**

**Given the two integers m and n, return the number of possible unique paths that the robot can take to reach the bottom-right corner.**

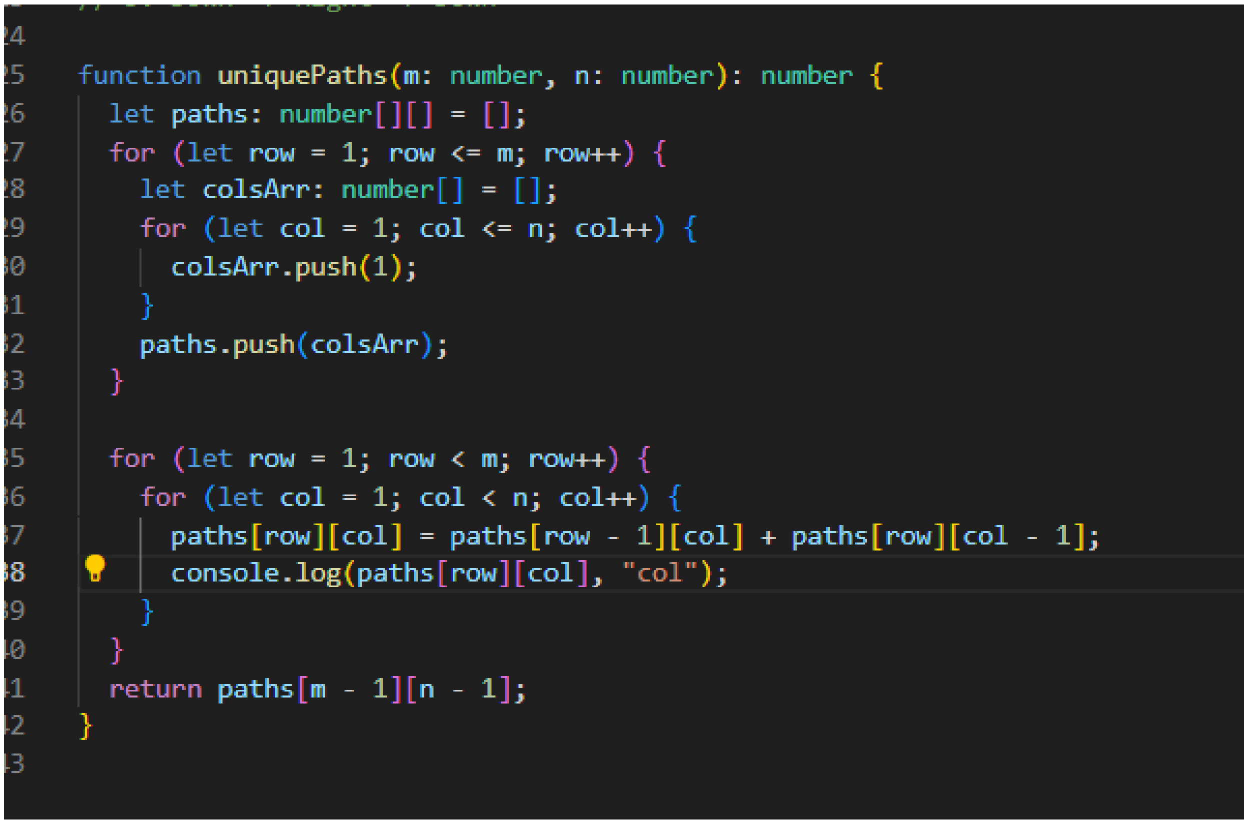
**The test cases are generated so that the answer will be less than or equal to 2 * 109.**

**Example 1:**

**Input: m = 3, n = 7**
**Output: 28**
**Example 2:**

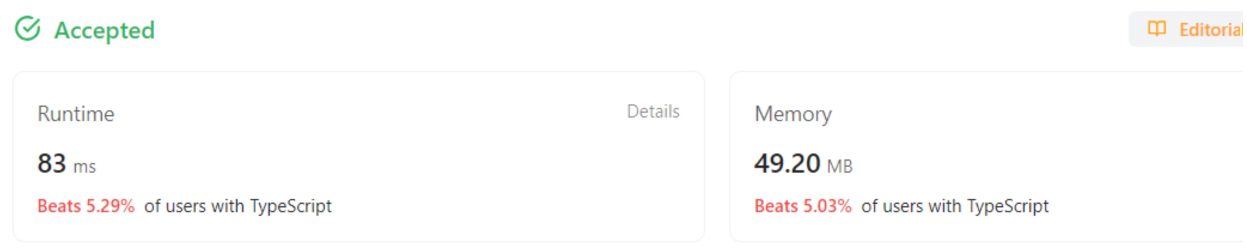
**Input: m = 3, n = 2**
**Output: 3**
**Explanation: From the top-left corner, there are a total of 3 ways to reach the bottom-right corner:**
**1. Right -> Down -> Down**
**2. Down -> Down -> Right**

## 3. Down -> Right -> Down



```
function uniquePaths(m: number, n: number): number {
  let paths: number[][] = [];
  for (let row = 1; row <= m; row++) {
    let colsArr: number[] = [];
    for (let col = 1; col <= n; col++) {
      colsArr.push(1);
    }
    paths.push(colsArr);
  }

  for (let row = 1; row < m; row++) {
    for (let col = 1; col < n; col++) {
      paths[row][col] = paths[row - 1][col] + paths[row][col - 1];
      console.log(paths[row][col], "col");
    }
  }
  return paths[m - 1][n - 1];
}
```



Accepted

Runtime

83 ms

Beats 5.29% of users with TypeScript

Memory

49.20 MB

Beats 5.03% of users with TypeScript