

Unique Paths

Grid DP — count paths from top-left to bottom-right ■

● MEDIUM

■ Hook

A robot on an $m \times n$ grid starts top-left, moves only right or down, reaches bottom-right. How many unique paths? Classic 2D DP — every cell's count = paths from above + paths from left.

■ DP Grid — 3×3

```
dp[i][j] = paths to reach cell (i,j)
First row: all 1s (can only come from left)
First col: all 1s (can only come from above)
[1][1][1]
[1][2][3]
[1][3][6] ← dp[2][2]=6 = answer
dp[i][j] = dp[i-1][j] + dp[i][j-1]
```

■ Code

```
public int uniquePaths(int m, int n) {
    int[][] dp = new int[m][n];
    for (int i=0; i<m; i++) dp[i][0]=1; // first col
    for (int j=0; j<n; j++) dp[0][j]=1; // first row
    for (int i=1; i<m; i++)
        for (int j=1; j<n; j++)
            dp[i][j] = dp[i-1][j] + dp[i][j-1];
    return dp[m-1][n-1];
}
// Math: C(m+n-2, m-1) — combinatorics solution O(1) space
```

■ TIME

$O(m \times n)$

■ SPACE

$O(m \times n) \rightarrow O(n)$ with 1D

■ Practice

#	Problem	Diff	Pattern
---	---------	------	---------

1	Unique Paths (LC 62)	Medium	Grid DP
2	Unique Paths II (LC 63)	Medium	Grid DP with obstacles
3	Minimum Path Sum (LC 64)	Medium	Grid DP min
4	Triangle (LC 120)	Medium	Triangle DP
5	Dungeon Game (LC 174)	Hard	Reverse Grid DP

- ✓ $dp[i][j] = dp[i-1][j] + dp[i][j-1]$
- ✓ Base: first row and first column are all 1
- ✓ Space optimize to 1D array
- ✓ Math formula: $C(m+n-2, m-1)$ — no DP needed
- ✓ Time: $O(mn)$ | Space: $O(mn)$ or $O(n)$

■ MENTAL MODEL — Remember This Forever

Every cell in the grid can only be reached from above or from the left. So the number of ways to reach it = sum of ways to reach those two neighbors. The grid fills like Pascal's triangle — each cell is the sum of its two parents.