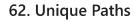


□ Discuss (999+)



6



Description

6 Solution

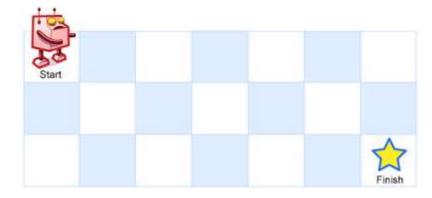
There is a robot on an $m \times n$ grid. The robot is initially located at the **top-left corner** (i.e., grid[0][0]). The robot tries to move to the **bottom-right corner** (i.e., grid[m - 1][n - 1]). The robot can only move either down or right at any point in time.

Submissions

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 $2 * 10^9$.

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

Constraints:

≡ Problems

➢ Pick One

< Prev

62/2184 Next >

Accepted

Your input

Output

Expected

Testcase Run Code Result Debugger 🔓

3

28

28

Runtime: 10 ms

▶ Run Coo

► Run Code ^ Submit

Diff

	i Java		$lacktriangledown$ Autocomplete i {} \circlearrowleft \odot \Box
	1	•	<pre>class Solution {</pre>
	2		
	3	•	<pre>private int uniquePaths(int m, int n, HashMap<string, integer=""> hashMap) {</string,></pre>
	4		
	5		<pre>String key = String.valueOf(m) + ',' + n;</pre>
	6		
	7		if (m == 1 && n == 1)
	8		return 1;
	9		else if $(m == 0 \mid \mid n == 0)$
	10		return 0;
	11		<pre>else if (hashMap.containsKey(key))</pre>
	12		return hashMap.get(key);
	13	•	else {
	14		hashMap.put(key, uniquePaths(m - 1, n, hashMap) +
			<pre>uniquePaths(m, n - 1, hashMap));</pre>
	15		<pre>return hashMap.get(key);</pre>
	16		}
	17		}
•	18		
	19	•	<pre>public int uniquePaths(int m, int n) {</pre>
	20		
	21		<pre>HashMap<string, integer=""> hashMap = new HashMap<>();</string,></pre>
	22		
	23		<pre>return uniquePaths(m, n, hashMap);</pre>
	24		}
	25		}