(/problems د o subseauer

# 6110. Number of Increasing Paths in a Grid

My Submissions (/contest/weekly-contest-300/problems/number-of-increasing-paths-in-a-grid/submissions/)

You are given an m x n integer matrix grid, where you can move from a cell to any adjacent cell in all 4 directions.

Return the number of strictly increasing paths in the grid such that you can start from any cell and end at any cell. Since the

answer may be very large, return it **modulo**  $10^9 + 7$ .

Two paths are considered different if they do not have exactly the same sequence of visited cells.

# User Accepted: 0 User Tried: 0 Total Accepted: 0 Total Submissions: 0 Difficulty: (Hard)

### Example 1:

1	1
3	4

```
Input: grid = [[1,1],[3,4]]
Output: 8
Explanation: The strictly increasing paths are:
    Paths with length 1: [1], [1], [3], [4].
    Paths with length 2: [1 -> 3], [1 -> 4], [3 -> 4].
    Paths with length 3: [1 -> 3 -> 4].
The total number of paths is 4 + 3 + 1 = 8.
```

## Example 2:

```
Input: grid = [[1],[2]]
Output: 3
Explanation: The strictly increasing paths are:
    Paths with length 1: [1], [2].
    Paths with length 2: [1 -> 2].
The total number of paths is 2 + 1 = 3.
```

### **Constraints:**

```
    m == grid.length
    n == grid[i].length
    1 <= m, n <= 1000</li>
    1 <= m * n <= 10<sup>5</sup>
    1 <= grid[i][j] <= 10<sup>5</sup>
```

```
JavaScript •
```

```
const initialize2DArray = (n, m) \Rightarrow \{ let d = []; for (let i = 0; i < n; i++) \{ let t = Array(m).fill(0); d.push(t); \} \}
 2
 3
    const mod = 1e9 + 7, dx = [0, 0, -1, 1], dy = [-1, 1, 0, 0];
    const countPaths = (g) \Rightarrow \{
 4
        let n = g.length, m = g[0].length, d = [], dp = initialize2DArray(n, m), res = 0;
 5
6٠
         for (let i = 0; i < n; i++) {
 7
             for (let j = 0; j < m; j++) d.push([i, j, g[i][j]]);
 8
        d.sort((x, y) \Rightarrow x[2] - y[2]);
9
         for (const [x, y] of d) {
10
11
             dp[x][y]++;
             res += dp[x][y];
12
13
             res %= mod;
             for (let k = 0; k < 4; k++) {
14
15
                 let nx = x + dx[k], ny = y + dy[k];
                 if (nx < 0 \mid | nx >= n \mid | ny < 0 \mid | ny >= m) continue;
16
17
                 if (g[nx][ny] > g[x][y]) {
18
                      dp[nx][ny] += dp[x][y];
                      dp[nx][ny] \approx mod;
19
```

```
7/3/22, 2:00 AM
                                                                  Number of Increasing Paths in a Grid - LeetCode Contest
    20
                       }
    21
             }
    22
    23
             return res;
    24
  ☐ Custom Testcase
                          Use Example Testcases
                                                                                                                                        Run
                                                                                                                                                   △ Submit
  Submission Result: Accepted (/submissions/detail/737186214/) 2
                                                                                      More Details > (/submissions/detail/737186214/)
  Share your acceptance!
   Copyright © 2022 LeetCode
   Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)
   United States (/region)
```