(/problems

6053. Count Unquarded Cells in the Grid

My Submissions (/contest/biweekly-contest-77/problems/count-unguarded-cells-in-the-grid/submissions/)

Back to Contest (/contest/biweekly-contest-77/)

ref=nb_npl)

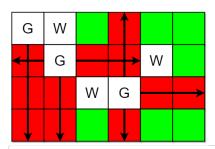
You are given two integers m and n representing a **0-indexed** m x n grid. You are also given two 2D integer arrays guards and walls where guards[i] = $[row_i, col_i]$ and walls[j] = $[row_j, col_j]$ represent the positions of the ith guard and jth wall respectively.

A guard can see **every** cell in the four cardinal directions (north, east, south, or west) starting from their position unless **obstructed** by a wall or another guard. A cell is **guarded** if there is **at least** one guard that can see it.

Return the number of unoccupied cells that are not guarded.

Total Submissions:	0
Total Accepted:	0
User Tried:	0
User Accepted:	0

Example 1:

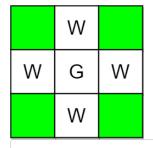


Input: m = 4, n = 6, guards = [[0,0],[1,1],[2,3]], walls = [[0,1],[2,2],[1,4]]
Output: 7

Explanation: The guarded and unguarded cells are shown in red and green respectively in the above diagram.

There are a total of 7 unguarded cells, so we return 7.

Example 2:



Input: m = 3, n = 3, guards = [[1,1]], walls = [[0,1],[1,0],[2,1],[1,2]]

Output: 4

Explanation: The unguarded cells are shown in green in the above diagram.

There are a total of 4 unguarded cells, so we return 4.

Constraints:

- 1 <= m, n <= 10⁵
- $2 \le m * n \le 10^5$
- 1 <= guards.length, walls.length <= $5 * 10^4$
- 2 <= guards.length + walls.length <= m * n
- guards[i].length == walls[j].length == 2
- 0 <= row_i, row_i < m
- $0 \le \operatorname{col}_i$, $\operatorname{col}_i < n$
- All the positions in guards and walls are unique.



```
for (int[] wall : walls) visit[wall[0]][wall[1]] = 'W';
 6
 7
             for (int[] guard : guards) visit[guard[0]][guard[1]] = 'G';
             for (int[] guard : guards) {
 8 ▼
 9
                 int x = guard[0], y = guard[1];
                 for (int j = y + 1; j < m; j++) { // right
10 ▼
                     if (hit(visit[x][j])) break;
11
12
                     visit[x][j] = 'R';
13
14 ▼
                 for (int j = y - 1; j >= 0; j--) { // left
                     if (hit(visit[x][j])) break;
15
                     visit[x][j] = 'R';
16
17
                 for (int i = x + 1; i < n; i++) { // down
18 ▼
                     if (hit(visit[i][y])) break;
19
20
                     visit[i][y] = 'R';
21
22 •
                 for (int i = x - 1; i >= 0; i--) { // up
                     if (hit(visit[i][y])) break;
23
24
                     visit[i][y] = 'R';
25
                 }
26
             }
27
             // tr(visit);
             for (int i = 0; i < n; i++) {
28 ▼
                 for (int j = 0; j < m; j++) {
29 ▼
                     if (visit[i][j] == 0) cnt++;
30
31
32
33
             return cnt;
34
        }
35
36 ▼
        boolean hit(char cur) {
             if (cur == 'W') return true;
if (cur == 'G') return true;
37
38
39
             return false;
40
        }
41
    }
```

☐ Custom Testcase

Use Example Testcases

Submission Result: Accepted (/submissions/detail/690321046/) ?

More Details > (/submissions/detail/690321046/)

Run

Submit
 Su

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)