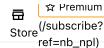
(/) Explore

Problems(/problemset/all/)

Interview

(/contest/) Contest

Discuss(/discuss/)









5815. Maximum Number of Points with Cost

My Submissions (/contest/weekly-contest-250/problems/maximum-number-of-points-with-cost/submissions/)

Back to Contest (/contest/weekly-contest-250/)

You are given an $m \times n$ integer matrix points (**0-indexed**). Starting with 0 points, you want to **maximize** the number of points you can get from the matrix.

To gain points, you must pick one cell in **each row**. Picking the cell at coordinates (r, c) will **add** points[r][c] to your score.

However, you will lose points if you pick a cell too far from the cell that you picked in the previous row. For every two adjacent rows $\, r \,$ and $\, r \, + \, 1 \,$ (where $\, 0 \, <= \, r \, < \, m \, - \, 1 \,$), picking cells at coordinates $\, (\, r \, , \, c_1 \,) \,$ and $\, (\, r \, + \, 1 \, , \, c_2 \,) \,$ will **subtract** abs($\, c_1 \, - \, c_2 \,) \,$ from your score.

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

Return the **maximum** number of points you can achieve.

abs(x) is defined as:

- x for x >= 0.
- -x for x < 0.

Example 1:

1	2	3
1	5	1
3	1	1

Input: points = [[1,2,3],[1,5,1],[3,1,1]]

Output: 9
Explanation:

The blue cells denote the optimal cells to pick, which have coordinates (0, 2), (1, 1), and (2, 1), and (3, 1) are coordinates (0, 2), (1, 1), and (2, 1) and (3, 1) are coordinates (0, 2), (1, 1), and (2, 1) are coordinates (3, 2), (3, 1), and (3, 1) are coordinates (3, 2), (3, 2), and (3, 2), and (3, 2), are coordinates (3, 2), are coordinates (3, 2), and (3, 2), are coordinates (3, 2), are coordinates (3, 2), and (3, 2), are coordinates (3, 2), are coordinates

However, you must subtract abs(2-1) + abs(1-0) = 2 from your score.

Your final score is 11 - 2 = 9.

Example 2:

1	5
2	3
4	2

```
Input: points = [[1,5],[2,3],[4,2]]
Output: 11
Explanation:
The blue cells denote the optimal cells to pick, which have coordinates (0, 1), (1, 1), and (2, 1))
You add (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) and (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1) and (5, 1) are initial score is (5, 1).
```

Constraints:

```
    m == points.length
    n == points[r].length
    1 <= m, n <= 10<sup>5</sup>
    1 <= m * n <= 10<sup>5</sup>
    0 <= points[r][c] <= 10<sup>5</sup>
```

```
JavaScript ▼
```

```
const abs = Math.abs;
 2 \vee const maxPoints = (g) => {
 3
        let n = g.length;
        let m = g[0].length;
 4
 5
        let dp = initialize2DArrayNew(n, m);
 6 ▼
        for (let i = 0; i < n; i++) {
 7 ▼
            for (let j = 0; j < m; j++) {
                 if (i == 0) {
 8 ▼
 9
                     dp[i][j] = g[i][j];
10 ▼
                 } else {
                     let max = 0;
11
12 ▼
                     for (let col = 0; col < m; col++) {
13
                         // pr()
                         // pr("row", i);
14
                         let rowDiff = abs(j - col);
15
16
                         let plus = g[i][j] + dp[i - 1][col] - rowDiff;
17
                         // pr(g[i][j], dp[i - 1][col], rowDiff);
18
                         max = Math.max(max, plus)
19
                     }
```

```
20
                     // pr("max", max);
21
                     dp[i][j] += max;
                 }
22
23
            }
24
        }
25
        // pr(dp);
26
        return Math.max.apply(Math, dp[n - 1]);
27
    };
28
    const initialize2DArrayNew = (n, m) \Rightarrow \{ let data = []; for (let i = 0; i < n; i++) \} 
29
    tmp = Array(m).fill(0); data.push(tmp); } return data; };
```

☐ Custom Testcase

Use Example Testcases





Submission Result: Accepted (/submissions/detail/524235915/) ?

More Details > (/submissions/detail/524235915/)

Share your acceptance!

Copyright © 2021 LeetCode

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) |



United States (/region)