598 Range Addition II

2018年4月16日 18:11

Given an m * n matrix **M** initialized with all **0**'s and several update operations.

Operations are represented by a 2D array, and each operation is represented by an array with two **positive** integers **a** and **b**, which means **M[i][j]** should be **added by one** for all **0 <= i < a** and **0 <= j < b**.

You need to count and return the number of maximum integers in the matrix after performing all the operations.

Example 1:

Input:

m = 3, n = 3

operations = [[2,2],[3,3]]

Output: 4

Explanation:

```
Initially, M =
[[0, 0, 0],
 [0, 0, 0],
 [0, 0, 0]]
After performing [2,2], M =
[[1, 1, 0],
 [1, 1, 0],
 [0, 0, 0]]
After performing [3,3], M =
[[2, 2, 1],
 [2, 2, 1],
 [1, 1, 1]]
```

So the maximum integer in M is 2, and there are four of it in M. So return 4.

Note:

1. The range of m and n is [1,40000].
2. The range of a is [1,m], and the range of b is [1,n].
3. The range of operations size won't exceed 10,000.

来自 <https://leetcode.com/problems/range-addition-ii/description/>

给定一个初始元素全部为 **0**，大小为 m*n 的矩阵 **M** 以及在 **M** 上的一系列更新操作。

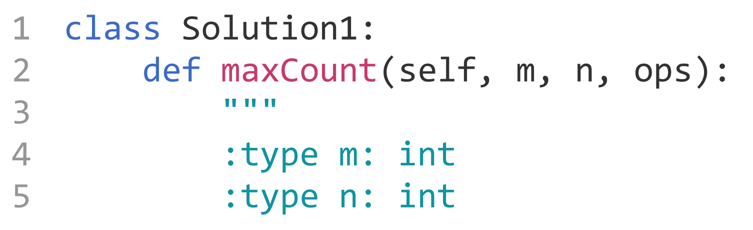
操作用二维数组表示，其中的每个操作用一个含有两个**正整数 a** 和 **b** 的数组表示，含义是将所有符合 **0 <= i < a** 以及 **0 <= j < b** 的元素 **M[i][j]** 的值都**增加 1**。

在执行给定的一系列操作后，你需要返回矩阵中含有最大整数的元素个数。

注意:

1. m 和 n 的范围是 [1,40000]。
2. a 的范围是 [1,m]，b 的范围是 [1,n]。
3. 操作数目不超过 10000。

Solution for Python3:



```python
class Solution1:
    def maxCount(self, m, n, ops):
        """
        :type m: int
        :type n: int
        :type ops: List[List[int]]
        :rtype: int
        """
        if not ops:
           return m * n
        m, n = [min(i) for i in zip(*ops)]

        return m * n

class Solution2:
    def maxCount(self, m, n, ops):
        """
        :type m: int
        :type n: int
        :type ops: List[List[int]]
        :rtype: int
        """
        if not ops:
           return m * n
        return min(op[0] for op in ops) * min(op[1] for op in ops)
```

Solution for C++:

```
class Solution {
public:
    int maxCount(int m, int n, vector<vector<int>>& ops) {
        for (auto op : ops) {
            m = min(m, op[0]);
            n = min(n, op[1]);
        }
        return m * n;
    }
};
```