1351. Count Negative Numbers in a Sorted Matrix

Solved

Easy | Topics | Companies | Hint

Given a `m x n` matrix `grid` which is sorted in non-increasing order both row-wise and column-wise, return *the number of* ***negative*** *numbers in* `grid`.

Example 1:

```
Input: grid = [[4,3,2,-1],[3,2,1,-1],[1,1,-1,-2],[-1,-1,-2,-3]]
Output: 8
Explanation: There are 8 negatives number in the matrix.
```

Example 2:

```
Input: grid = [[3,2],[1,0]]
Output: 0
```

Constraints:

- `m == grid.length`
- `n == grid[i].length`
- `1 <= m, n <= 100`
- `-100 <= grid[i][j] <= 100`

Follow up: Could you find an `O(n + m)` solution?

Python:

```
class Solution:
    def countNegatives(self, grid: List[List[int]]) -> int:
        m = len(grid)
        n = len(grid[0])

        i = m - 1
        j = 0
        res = 0
```

```
        while i >= 0 and j < n:
            if grid[i][j] < 0:
                res += n - j
                i -= 1
            else:
                j += 1

        return res
```

JavaScript:

```
const countNegatives = grid => {
    const m = grid.length, n = grid[0].length;
    let i = m - 1, j = 0;

    let res = 0;

    while (i >= 0 && j < n) {
        if (grid[i][j] < 0) {
            res += n - j;
            i--;
        } else {
            j++;
        }
    }

    return res;
};
```

Java:

```
class Solution {
    public int countNegatives(int[][] grid) {
        int m = grid.length, n = grid[0].length;
        int i = m - 1, j = 0;

        int res = 0;

        while (i >= 0 && j < n) {
            if (grid[i][j] < 0) {
                res += n - j;
                i--;
            } else {
                j++;
            }
        }
```

```
        return res;
    }
}
```