Problem 2713: Maximum Strictly Increasing Cells in a Matrix

Problem Information

Difficulty: Hard
Acceptance Rate: 31.14%
Paid Only: No
Tags: Array, Hash Table, Binary Search, Dynamic Programming, Memoization, Sorting, Matrix, Ordered Set

Problem Description

Given a **1-indexed** `m x n` integer matrix `mat`, you can select any cell in the matrix as your **starting cell**.

From the starting cell, you can move to any other cell **in the** **same row or column** , but only if the value of the destination cell is **strictly greater** than the value of the current cell. You can repeat this process as many times as possible, moving from cell to cell until you can no longer make any moves.

Your task is to find the **maximum number of cells** that you can visit in the matrix by starting from some cell.

Return _an integer denoting the maximum number of cells that can be visited._

Example 1:

![](https://assets.leetcode.com/uploads/2023/04/23/diag1drawio.png)

Input: mat = [[3,1],[3,4]] **Output:** 2 **Explanation:** The image shows how we can visit 2 cells starting from row 1, column 2. It can be shown that we cannot visit more than 2 cells no matter where we start from, so the answer is 2.

Example 2:

![](https://assets.leetcode.com/uploads/2023/04/23/diag3drawio.png)

Input: mat = [[1,1],[1,1]] **Output:** 1 **Explanation:** Since the cells must be strictly increasing, we can only visit one cell in this example.

Example 3:

![](https://assets.leetcode.com/uploads/2023/04/23/diag4drawio.png)

Input: mat = [[3,1,6],[-9,5,7]] **Output:** 4 **Explanation:** The image above shows how we can visit 4 cells starting from row 2, column 1. It can be shown that we cannot visit more than 4 cells no matter where we start from, so the answer is 4.

Constraints:

* `m == mat.length ` * `n == mat[i].length ` * `1 <= m, n <= 105` * `1 <= m * n <= 105` * `-105 <= mat[i][j] <= 105`

Code Snippets

C++:

```
class Solution {
public:
int maxIncreasingCells(vector<vector<int>>& mat) {

}
};
```

Java:

```
class Solution {
public int maxIncreasingCells(int[][] mat) {

}
}
```

Python3:

```
class Solution:
def maxIncreasingCells(self, mat: List[List[int]]) -> int:
```