Problem 3193: Count the Number of Inversions

Problem Information

Difficulty: Hard
Acceptance Rate: 28.91%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You are given an integer `n` and a 2D array `requirements`, where `requirements[i] = [endi, cnti]` represents the end index and the **inversion** count of each requirement.

A pair of indices `(i, j)` from an integer array `nums` is called an **inversion** if:

* `i < j` and `nums[i] > nums[j]`

Return the number of permutations `perm` of `[0, 1, 2, ..., n - 1]` such that for **all** `requirements[i]`, `perm[0..endi]` has exactly `cnti` inversions.

Since the answer may be very large, return it **modulo** `109 + 7`.

Example 1:

Input: n = 3, requirements = [[2,2],[0,0]]

Output: 2

Explanation:

The two permutations are:

* `[2, 0, 1]` * Prefix `[2, 0, 1]` has inversions `(0, 1)` and `(0, 2)`. * Prefix `[2]` has 0 inversions.
* `[1, 2, 0]` * Prefix `[1, 2, 0]` has inversions `(0, 2)` and `(1, 2)`. * Prefix `[1]` has 0 inversions.

Example 2:

Input: n = 3, requirements = [[2,2],[1,1],[0,0]]

Output: 1

Explanation:

The only satisfying permutation is `[2, 0, 1]`:

* Prefix `[2, 0, 1]` has inversions `(0, 1)` and `(0, 2)`. * Prefix `[2, 0]` has an inversion `(0, 1)`. * Prefix `[2]` has 0 inversions.

Example 3:

Input: n = 2, requirements = [[0,0],[1,0]]

Output: 1

Explanation:

The only satisfying permutation is `[0, 1]`:

* Prefix `[0]` has 0 inversions. * Prefix `[0, 1]` has an inversion `(0, 1)`.

Constraints:

* `2 <= n <= 300` * `1 <= requirements.length <= n` * `requirements[i] = [endi, cnti]` * `0 <= endi <= n - 1` * `0 <= cnti <= 400` * The input is generated such that there is at least one `i` such that `endi == n - 1`. * The input is generated such that all `endi` are unique.

Code Snippets

C++:

```
class Solution {
public:
int numberOfPermutations(int n, vector<vector<int>>& requirements) {
```

```
}
};
```

Java:

```
class Solution {
public int numberOfPermutations(int n, int[][] requirements) {

}
}
```

Python3:

```
class Solution:
def numberOfPermutations(self, n: int, requirements: List[List[int]]) -> int:
```