Problem 673: Number of Longest Increasing Subsequence

Problem Information

Difficulty: Medium
Acceptance Rate: 50.89%
Paid Only: No
Tags: Array, Dynamic Programming, Binary Indexed Tree, Segment Tree

Problem Description

Given an integer array `nums`, return _the number of longest increasing subsequences._

Notice that the sequence has to be **strictly** increasing.

Example 1:

Input: nums = [1,3,5,4,7] **Output:** 2 **Explanation:** The two longest increasing subsequences are [1, 3, 4, 7] and [1, 3, 5, 7].

Example 2:

Input: nums = [2,2,2,2,2] **Output:** 5 **Explanation:** The length of the longest increasing subsequence is 1, and there are 5 increasing subsequences of length 1, so output 5.

Constraints:

* `1 <= nums.length <= 2000` * `-106 <= nums[i] <= 106` * The answer is guaranteed to fit inside a 32-bit integer.

Code Snippets

C++:

```
class Solution {
public:
int findNumberOfLIS(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int findNumberOfLIS(int[] nums) {

}
}
```

Python3:

```
class Solution:
def findNumberOfLIS(self, nums: List[int]) -> int:
```