Problem 300: Longest Increasing Subsequence

Problem Information

Difficulty: Medium
Acceptance Rate: 58.67%
Paid Only: No
Tags: Array, Binary Search, Dynamic Programming

Problem Description

Given an integer array `nums`, return _the length of the longest**strictly increasing**_ _**subsequence**_.

Example 1:

Input: nums = [10,9,2,5,3,7,101,18] **Output:** 4 **Explanation:** The longest increasing subsequence is [2,3,7,101], therefore the length is 4.

Example 2:

Input: nums = [0,1,0,3,2,3] **Output:** 4

Example 3:

Input: nums = [7,7,7,7,7,7,7] **Output:** 1

Constraints:

* `1 <= nums.length <= 2500` * `-104 <= nums[i] <= 104`

Follow up: Can you come up with an algorithm that runs in `O(n log(n))` time complexity?

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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