

Problem 1964: Find the Longest Valid Obstacle Course at Each Position

Problem Information

Difficulty: Hard

Acceptance Rate: 62.55%

Paid Only: No

Tags: Array, Binary Search, Binary Indexed Tree

Problem Description

You want to build some obstacle courses. You are given a **0-indexed** integer array `obstacles` of length `n`, where `obstacles[i]` describes the height of the `ith` obstacle.

For every index `i` between `0` and `n - 1` (**inclusive**), find the length of the **longest obstacle course** in `obstacles` such that:

- * You choose any number of obstacles between `0` and `i` **inclusive**.
- * You must include the `ith` obstacle in the course.
- * You must put the chosen obstacles in the **same order** as they appear in `obstacles`.
- * Every obstacle (except the first) is **taller** than or the **same height** as the obstacle immediately before it.

Return _an array_ `ans` _of length_ `n`, _where_ `ans[i]` _is the length of the**longest obstacle course** for index_ `i` _as described above_.

Example 1:

Input: obstacles = [1,2,3,2] **Output:** [1,2,3,3] **Explanation:** The longest valid obstacle course at each position is: - i = 0: [1], [1] has length 1. - i = 1: [1, 2], [1,2] has length 2. - i = 2: [1, 2, 3], [1,2,3] has length 3. - i = 3: [1, 2, 3, 2], [1,2,2] has length 3.

Example 2:

Input: obstacles = [2,2,1] **Output:** [1,2,1] **Explanation:** The longest valid obstacle course at each position is: - i = 0: [2], [2] has length 1. - i = 1: [2, 2], [2,2] has length 2. - i = 2: [2,2,1], [1] has length 1.

****Example 3:****

****Input:**** obstacles = [3,1,5,6,4,2] ****Output:**** [1,1,2,3,2,2] ****Explanation:**** The longest valid obstacle course at each position is: - i = 0: [3], [3] has length 1. - i = 1: [3,1], [1] has length 1. - i = 2: [3,1,5], [3,5] has length 2. [1,5] is also valid. - i = 3: [3,1,5,6], [3,5,6] has length 3. [1,5,6] is also valid. - i = 4: [3,1,5,6,4], [3,4] has length 2. [1,4] is also valid. - i = 5: [3,1,5,6,4,2], [1,2] has length 2.

****Constraints:****

```
* `n == obstacles.length` * `1 <= n <= 105` * `1 <= obstacles[i] <= 107`
```

Code Snippets

C++:

```
class Solution {
public:
    vector<int> longestObstacleCourseAtEachPosition(vector<int>& obstacles) {
        }
    };
}
```

Java:

```
class Solution {
public int[] longestObstacleCourseAtEachPosition(int[] obstacles) {
    }
}
```

Python3:

```
class Solution:
    def longestObstacleCourseAtEachPosition(self, obstacles: List[int]) ->
        List[int]:
```