Problem 1944: Number of Visible People in a Queue

Problem Information

Difficulty: Hard
Acceptance Rate: 71.98%
Paid Only: No
Tags: Array, Stack, Monotonic Stack

Problem Description

There are `n` people standing in a queue, and they numbered from `0` to `n - 1` in **left to right** order. You are given an array `heights` of **distinct** integers where `heights[i]` represents the height of the `ith` person.

A person can **see** another person to their right in the queue if everybody in between is **shorter** than both of them. More formally, the `ith` person can see the `jth` person if `i < j` and `min(heights[i], heights[j]) > max(heights[i+1], heights[i+2], ..., heights[j-1])`.

Return _an array_`answer` _of length_`n` _where_`answer[i]`_is the**number of people** the _`ith` _person can**see** to their right in the queue_.

Example 1:

Input: heights = [10,6,8,5,11,9] **Output:** [3,1,2,1,1,0] **Explanation:** Person 0 can see person 1, 2, and 4. Person 1 can see person 2. Person 2 can see person 3 and 4. Person 3 can see person 4. Person 4 can see person 5. Person 5 can see no one since nobody is to the right of them.

Example 2:

Input: heights = [5,1,2,3,10] **Output:** [4,1,1,1,0]

Constraints:

* `n == heights.length` * `1 <= n <= 105` * `1 <= heights[i] <= 105` * All the values of `heights` are **unique**.

Code Snippets

C++:

```
class Solution {
public:
vector<int> canSeePersonsCount(vector<int>& heights) {

}
};
```

Java:

```
class Solution {
public int[] canSeePersonsCount(int[] heights) {

}
}
```

Python3:

```
class Solution:
def canSeePersonsCount(self, heights: List[int]) -> List[int]:
```