# 2832. Maximal Range That Each Element Is Maximum in It Premium

Medium | Topics | Companies | Hint

You are given a **0-indexed** array `nums` of **distinct** integers.

Let us define a **0-indexed** array `ans` of the same length as `nums` in the following way:

- `ans[i]` is the **maximum** length of a subarray `nums[l..r]`, such that the maximum element in that subarray is equal to `nums[i]`.

Return *the array* `ans`.

**Note** that a **subarray** is a contiguous part of the array.

**Example 1:**

```
Input: nums = [1,5,4,3,6]
Output: [1,4,2,1,5]
Explanation: For nums[0] the longest subarray in which 1 is the maximum is nums[0..0] so ans[0] = 1.
For nums[1] the longest subarray in which 5 is the maximum is nums[0..3] so ans[1] = 4.
For nums[2] the longest subarray in which 4 is the maximum is nums[2..3] so ans[2] = 2.
For nums[3] the longest subarray in which 3 is the maximum is nums[3..3] so ans[3] = 1.
For nums[4] the longest subarray in which 6 is the maximum is nums[0..4] so ans[4] = 5.
```

**Example 2:**

```
Input: nums = [1,2,3,4,5]
Output: [1,2,3,4,5]
Explanation: For nums[i] the longest subarray in which it's the maximum is nums[0..i] so ans[i] = i + 1.
```

**Constraints:**

- `1 <= nums.length <= 10^5`
- `1 <= nums[i] <= 10^5`
- All elements in `nums` are distinct.

Seen this question in a real interview before? 1/5

Yes No

Accepted 1.5K | Submissions 2.1K | Acceptance Rate 72.2%

Topics

Array Stack Monotonic Stack

Companies

0 - 6 months

Amazon 2

Hint 1

For each index, we must find the nearest bigger element on both its left and right sides.

Hint 2

First, find the nearest bigger element on the left side of each element. To do that, use a stack of pairs `(value, index)`.

Hint 3

Start iterating from the beginning of the array.

Hint 4

Whenever we reach an element `nums[index]`, while the top of the stack is smaller than `nums[index]`, we pop from the stack.

Hint 5

If there is an element left in the stack, `top.index + 1` would be the answer. Otherwise, `0` is the answer.

Hint 6

After that, we push `(nums[index], index)` to the stack and go for the next element.

Discussion (5)