3254. Find the Power of K-Size Subarrays I

Medium

Topics

Companies

Hint

You are given an array of integers `nums` of length `n` and a *positive* integer `k`.

The **power** of an array is defined as:

- Its **maximum** element if *all* of its elements are **consecutive** and **sorted** in **ascending** order.
- -1 otherwise.

You need to find the **power** of all

subarrays

of `nums` of size `k`.

Return an integer array `results` of size `n - k + 1`, where `results[i]` is the *power* of `nums[i..(i + k - 1)]`.

Example 1:

Input: nums = [1,2,3,4,3,2,5], k = 3

Output: [3,4,-1,-1,-1]

Explanation:

There are 5 subarrays of `nums` of size 3:

- `[1, 2, 3]` with the maximum element 3.
- `[2, 3, 4]` with the maximum element 4.
- `[3, 4, 3]` whose elements are **not** consecutive.
- `[4, 3, 2]` whose elements are **not** sorted.
- `[3, 2, 5]` whose elements are **not** consecutive.

Example 2:

Input: nums = [2,2,2,2,2], k = 4

Output: [-1,-1]

Example 3:

Input: nums = [3,2,3,2,3,2], k = 2

Output: [-1,3,-1,3,-1]

Constraints:

- `1 <= n == nums.length <= 500`
- `1 <= nums[i] <= 105`
- `1 <= k <= n`

Solution:

```
class Solution {

    public int[] resultsArray(int[] nums, int k) {

        int arr[] = new int[nums.length - k+1];



        int i = 0; int j = k-1;

        int a = 0;

        while(j < nums.length){

            if(isSorted(nums, i, j)){

                arr[a] = nums [j];

            }else{

                arr[a] = -1;

            }

            i++;

            j++;

            a++;
        }

        return arr;

    }



    public boolean isSorted(int[] nums, int start, int end){

        for(int i = start; i < end; i++){

            if(nums[i] +1 != nums[i+1]){

                return false;

            }

        }

        return true;

    }

}
```