Problem 3636: Threshold Majority Queries

Problem Information

Difficulty: Hard
Acceptance Rate: 20.90%
Paid Only: No
Tags: Array, Hash Table, Binary Search, Divide and Conquer, Counting, Prefix Sum

Problem Description

You are given an integer array `nums` of length `n` and an array `queries`, where `queries[i] = [li, ri, thresholdi]`.

Return an array of integers `ans` where `ans[i]` is equal to the element in the subarray `nums[li...ri]` that appears **at least** `thresholdi` times, selecting the element with the **highest** frequency (choosing the **smallest** in case of a tie), or -1 if no such element _exists_.

Example 1:

Input: nums = [1,1,2,2,1,1], queries = [[0,5,4],[0,3,3],[2,3,2]]

Output: [1,-1,2]

Explanation:

Query | Sub-array | Threshold | Frequency table | Answer ---|---|---|---|--- [0, 5, 4] | [1, 1, 2, 2, 1, 1] | 4 | 1 -> 4, 2 -> 2 | 1 [0, 3, 3] | [1, 1, 2, 2] | 3 | 1 -> 2, 2 -> 2 | -1 [2, 3, 2] | [2, 2] | 2 | 2 -> 2 | 2
Example 2:

Input: nums = [3,2,3,2,3,2,3], queries = [[0,6,4],[1,5,2],[2,4,1],[3,3,1]]

Output: [3,2,3,2]

Explanation:

Query | Sub-array | Threshold | Frequency table | Answer ---|---|---|---|--- [0, 6, 4] | [3, 2, 3, 2, 3, 2, 3] | 4 | 3 -> 4, 2 -> 3 | 3 [1, 5, 2] | [2, 3, 2, 3, 2] | 2 | 2 -> 3, 3 -> 2 | 2 [2, 4, 1] | [3, 2, 3] | 1 | 3 -> 2, 2 -> 1 | 3 [3, 3, 1] | [2] | 1 | 2 -> 1 | 2

Constraints:

* `1 <= nums.length == n <= 104` * `1 <= nums[i] <= 109` * `1 <= queries.length <= 5 * 104` * `queries[i] = [li, ri, thresholdi]` * `0 <= li <= ri < n` * `1 <= thresholdi <= ri - li + 1`

Code Snippets

C++:

```
class Solution {
public:
vector<int> subarrayMajority(vector<int>& nums, vector<vector<int>>& queries)
{

}
};
```

Java:

```
class Solution {
public int[] subarrayMajority(int[] nums, int[][] queries) {

}
}
```

Python3:

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