Problem 3525: Find X Value of Array II

Problem Information

Difficulty: Hard
Acceptance Rate: 29.79%
Paid Only: No
Tags: Array, Math, Segment Tree

Problem Description

You are given an array of **positive** integers `nums` and a **positive** integer `k`. You are also given a 2D array `queries`, where `queries[i] = [indexi, valuei, starti, xi]`.

You are allowed to perform an operation **once** on `nums`, where you can remove any **suffix** from `nums` such that `nums` remains **non-empty**.

The **x-value** of `nums` **for a given** `x` is defined as the number of ways to perform this operation so that the **product** of the remaining elements leaves a _remainder_ of `x` **modulo** `k`.

For each query in `queries` you need to determine the **x-value** of `nums` for `xi` after performing the following actions:

* Update `nums[indexi]` to `valuei`. Only this step persists for the rest of the queries. * **Remove** the prefix `nums[0..(starti - 1)]` (where `nums[0..(-1)]` will be used to represent the **empty** prefix).

Return an array `result` of size `queries.length` where `result[i]` is the answer for the `ith` query.

A **prefix** of an array is a subarray that starts from the beginning of the array and extends to any point within it.

A **suffix** of an array is a subarray that starts at any point within the array and extends to the end of the array.

Note that the prefix and suffix to be chosen for the operation can be **empty**.

Note that x-value has a _different_ definition in this version.

Example 1:

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

Output: [2,2,2]

Explanation:

* For query 0, `nums` becomes `[1, 2, 2, 4, 5]`, and the empty prefix **must** be removed. The possible operations are: * Remove the suffix `[2, 4, 5]`. `nums` becomes `[1, 2]`. * Remove the empty suffix. `nums` becomes `[1, 2, 2, 4, 5]` with a product 80, which gives remainder 2 when divided by 3. * For query 1, `nums` becomes `[1, 2, 2, 3, 5]`, and the prefix `[1, 2, 2]` **must** be removed. The possible operations are: * Remove the empty suffix. `nums` becomes `[3, 5]`. * Remove the suffix `[5]`. `nums` becomes `[3]`. * For query 2, `nums` becomes `[1, 2, 2, 3, 5]`, and the empty prefix **must** be removed. The possible operations are: * Remove the suffix `[2, 2, 3, 5]`. `nums` becomes `[1]`. * Remove the suffix `[3, 5]`. `nums` becomes `[1, 2, 2]`.

Example 2:

Input: nums = [1,2,4,8,16,32], k = 4, queries = [[0,2,0,2],[0,2,0,1]]

Output: [1,0]

Explanation:

* For query 0, `nums` becomes `[2, 2, 4, 8, 16, 32]`. The only possible operation is: * Remove the suffix `[2, 4, 8, 16, 32]`. * For query 1, `nums` becomes `[2, 2, 4, 8, 16, 32]`. There is no possible way to perform the operation.

Example 3:

Input: nums = [1,1,2,1,1], k = 2, queries = [[2,1,0,1]]

Output: [5]

Constraints:

* `1 <= nums[i] <= 109` * `1 <= nums.length <= 105` * `1 <= k <= 5` * `1 <= queries.length <= 2 * 104` * `queries[i] == [indexi, valuei, starti, xi]` * `0 <= indexi <= nums.length - 1` * `1 <= valuei <= 109` * `0 <= starti <= nums.length - 1` * `0 <= xi <= k - 1`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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