Problem 3624: Number of Integers With Popcount-Depth Equal to K II

Problem Information

Difficulty: Hard
Acceptance Rate: 50.16%
Paid Only: No
Tags: Array, Divide and Conquer, Binary Indexed Tree, Segment Tree

Problem Description

You are given an integer array `nums`.

For any positive integer `x`, define the following sequence:

* `p0 = x` * `pi+1 = popcount(pi)` for all `i >= 0`, where `popcount(y)` is the number of set bits (1's) in the binary representation of `y`.

This sequence will eventually reach the value 1.

The **popcount-depth** of `x` is defined as the **smallest** integer `d >= 0` such that `pd = 1`.

For example, if `x = 7` (binary representation `"111"`). Then, the sequence is: `7 -> 3 -> 2 -> 1`, so the popcount-depth of 7 is 3.

You are also given a 2D integer array `queries`, where each `queries[i]` is either:

* `[1, l, r, k]` \- **Determine** the number of indices `j` such that `l <= j <= r` and the **popcount-depth** of `nums[j]` is equal to `k`. * `[2, idx, val]` \- **Update** `nums[idx]` to `val`.

Return an integer array `answer`, where `answer[i]` is the number of indices for the `ith` query of type `[1, l, r, k]`.

Example 1:

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

Output: [2,1]

Explanation:

`i` | `queries[i]` | `nums` | binary(`nums`) | popcount- depth | `[l, r]` | `k` | Valid `nums[j]` | updated `nums` | Answer ---|---|---|---|---|---|---|---|---|--- 0 | [1,0,1,1] | [2,4] | [10, 100] | [1, 1] | [0, 1] | 1 | [0, 1] | -- | 2 1 | [2,1,1] | [2,4] | [10, 100] | [1, 1] | -- | -- | -- | [2,1] | -- 2 | [1,0,1,0] | [2,1] | [10, 1] | [1, 0] | [0, 1] | 0 | [1] | -- | 1 Thus, the final `answer` is `[2, 1]`.

Example 2:

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

Output: [3,1,0]

Explanation:

`i` | `queries[i]` | `nums` | binary(`nums`) | popcount- depth | `[l, r]` | `k` | Valid `nums[j]` | updated `nums` | Answer ---|---|---|---|---|---|---|---|---|--- 0 | [1,0,2,2] | [3, 5, 6] | [11, 101, 110] | [2, 2, 2] | [0, 2] | 2 | [0, 1, 2] | -- | 3 1 | [2,1,4] | [3, 5, 6] | [11, 101, 110] | [2, 2, 2] | -- | -- | -- | [3, 4, 6] | -- 2 | [1,1,2,1] | [3, 4, 6] | [11, 100, 110] | [2, 1, 2] | [1, 2] | 1 | [1] | -- | 1 3 | [1,0,1,0] | [3, 4, 6] | [11, 100, 110] | [2, 1, 2] | [0, 1] | 0 | [] | -- | 0 Thus, the final `answer` is `[3, 1, 0]`.

Example 3:

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

Output: [1,0,1]

Explanation:

`i` | `queries[i]` | `nums` | binary(`nums`) | popcount- depth | `[l, r]` | `k` | Valid `nums[j]` | updated `nums` | Answer ---|---|---|---|---|---|---|---|---|--- 0 | [1,0,1,1] | [1, 2] | [1, 10] | [0, 1] | [0, 1] | 1 | [1] | -- | 1 1 | [2,0,3] | [1, 2] | [1, 10] | [0, 1] | -- | -- | -- | [3, 2] | 2 | [1,0,0,1] | [3, 2] | [11, 10] | [2, 1] | [0, 0] | 1 | [] | -- | 0 3 | [1,0,0,2] | [3, 2] | [11, 10] | [2, 1] | [0, 0] | 2 | [0] | -- | 1 Thus, the final `answer` is `[1, 0, 1]`.

Constraints:

* `1 <= n == nums.length <= 105` * `1 <= nums[i] <= 1015` * `1 <= queries.length <= 105` * `queries[i].length == 3` or `4` * `queries[i] == [1, l, r, k]` or, * `queries[i] == [2, idx, val]` * `0 <= l <= r <= n - 1` * `0 <= k <= 5` * `0 <= idx <= n - 1` * `1 <= val <= 1015`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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