2411. Smallest Subarrays With Maximum Bitwise OR

Description

You are given a **0-indexed** array `nums` of length `n`, consisting of non-negative integers. For each index `i` from `0` to `n - 1`, you must determine the size of the **minimum sized** non-empty subarray of `nums` starting at `i` (**inclusive**) that has the **maximum** possible **bitwise OR**.

- In other words, let B_{ij} be the bitwise OR of the subarray `nums[i...j]`. You need to find the smallest subarray starting at `i`, such that bitwise OR of this subarray is equal to $\max(B_{ik})$ where `i <= k <= n - 1`.

The bitwise OR of an array is the bitwise OR of all the numbers in it.

Return *an integer array* `answer` *of size* `n` *where* `answer[i]` *is the length of the* ***minimum*** *sized subarray starting at* `i` *with* ***maximum*** *bitwise OR.*

A **subarray** is a contiguous non-empty sequence of elements within an array.

Example 1:

```
 Input: nums = [1,0,2,1,3]
 Output: [3,3,2,2,1]
 Explanation:
The maximum possible bitwise OR starting at any index is 3.
- Starting at index 0, the shortest subarray that yields it is [1,0,2].
- Starting at index 1, the shortest subarray that yields the maximum bitwise OR is [0,2,1].
- Starting at index 2, the shortest subarray that yields the maximum bitwise OR is [2,1].
- Starting at index 3, the shortest subarray that yields the maximum bitwise OR is [1,3].
- Starting at index 4, the shortest subarray that yields the maximum bitwise OR is [3].
Therefore, we return [3,3,2,2,1].
```

Example 2:

```
 Input: nums = [1,2]
 Output: [2,1]
 Explanation:
Starting at index 0, the shortest subarray that yields the maximum bitwise OR is of length 2.
Starting at index 1, the shortest subarray that yields the maximum bitwise OR is of length 1.
Therefore, we return [2,1].
```

Constraints:

- `n == nums.length`
- $1 <= n <= 10^5$
- $0 <= nums[i] <= 10^9$