Problem 3095: Shortest Subarray With OR at Least K I

Problem Information

Difficulty: Easy
Acceptance Rate: 43.62%
Paid Only: No
Tags: Array, Bit Manipulation, Sliding Window

Problem Description

You are given an array `nums` of **non-negative** integers and an integer `k`.

An array is called **special** if the bitwise `OR` of all of its elements is **at least** `k`.

Return _the length of the**shortest** **special** **non-empty** subarray of_ `nums`, _or return_ `-1` _if no special subarray exists_.

Example 1:

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

Output: 1

Explanation:

The subarray `[3]` has `OR` value of `3`. Hence, we return `1`.

Note that `[2]` is also a special subarray.

Example 2:

Input: nums = [2,1,8], k = 10

Output: 3

Explanation:

The subarray `[2,1,8]` has `OR` value of `11`. Hence, we return `3`.

Example 3:

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

Output: 1

Explanation:

The subarray `[1]` has `OR` value of `1`. Hence, we return `1`.

Constraints:

* `1 <= nums.length <= 50` * `0 <= nums[i] <= 50` * `0 <= k < 64`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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