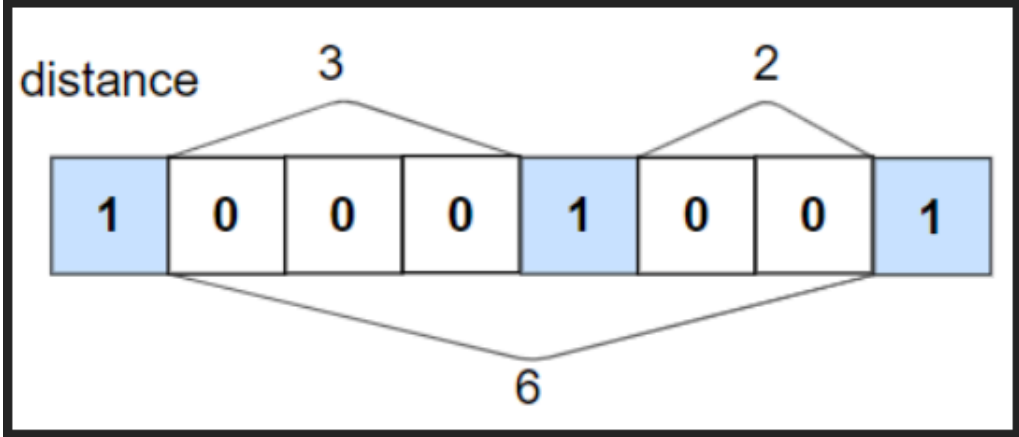
1437. Check If All 1's Are at Least Length K Places Away

Solved

Easy | Topics | Companies | Hint

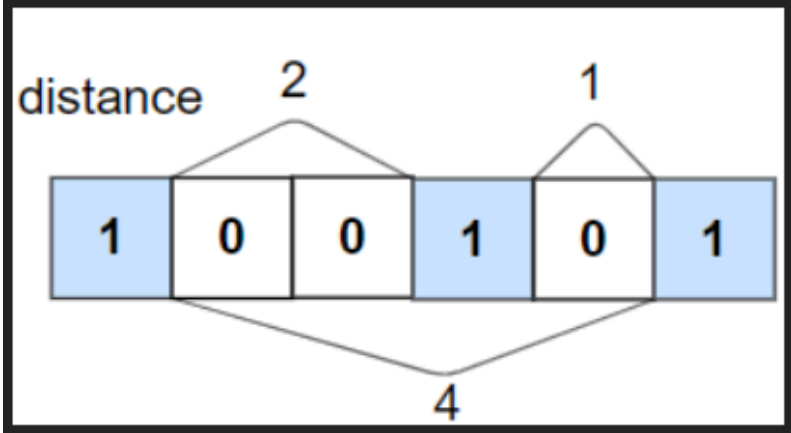
Given an binary array `nums` and an integer `k`, return `true` *if all* `1`*'s are at least* `k` *places away from each other, otherwise return* `false`.

Example 1:




```
Input: nums = [1,0,0,0,1,0,0,1], k = 2
Output: true
Explanation: Each of the 1s are at least 2 places away
from each other.
```

Example 2:




```
Input: nums = [1,0,0,1,0,1], k = 2
Output: false
Explanation: The second 1 and third 1 are only one apart
from each other.
```

Constraints:

- `1 <= nums.length <= 10^5`
- `0 <= k <= nums.length`
- `nums[i]` is `0` or `1`

Python:

```
class Solution:
    def kLengthApart(self, nums: List[int], k: int) -> bool:
        if k == 0:
            return True
        prev = None
        for i, num in enumerate(nums):
            if num == 1:
                if prev is not None and i - prev <= k:
                    return False
                prev = i
        return True
```

JavaScript:

```
function kLengthApart(nums, k) {
    let lastOneIndex = -1

    for (let i = 0; i < nums.length; i++) {
        if (nums[i] === 1) {
            if (lastOneIndex !== -1 && i - lastOneIndex - 1 < k) return false
            lastOneIndex = i
        }
    }

    return true
};
```

Java:

```
class Solution {
    public boolean kLengthApart(int[] nums, int k) {
        int lastIndex = -1;

        for (int i = 0; i < nums.length; i++) {

            if (nums[i] == 1) {
                if (lastIndex != -1 && i - lastIndex - 1 < k) {
                    return false;
                }
                lastIndex = i;
            }
        }

        return true;
    }
}
```