Problem 995: Minimum Number of K Consecutive Bit Flips

Problem Information

Difficulty: Hard
Acceptance Rate: 62.23%
Paid Only: No
Tags: Array, Bit Manipulation, Queue, Sliding Window, Prefix Sum

Problem Description

You are given a binary array `nums` and an integer `k`.

A **k-bit flip** is choosing a **subarray** of length `k` from `nums` and simultaneously changing every `0` in the subarray to `1`, and every `1` in the subarray to `0`.

Return _the minimum number of**k-bit flips** required so that there is no _`0` _in the array_. If it is not possible, return `-1`.

A **subarray** is a **contiguous** part of an array.

Example 1:

Input: nums = [0,1,0], k = 1 **Output:** 2 **Explanation:** Flip nums[0], then flip nums[2].

Example 2:

Input: nums = [1,1,0], k = 2 **Output:** -1 **Explanation:** No matter how we flip subarrays of size 2, we cannot make the array become [1,1,1].

Example 3:

Input: nums = [0,0,0,1,0,1,1,0], k = 3 **Output:** 3 **Explanation:** Flip nums[0],nums[1],nums[2]: nums becomes [1,1,1,1,0,1,1,0] Flip nums[4],nums[5],nums[6]: nums becomes [1,1,1,1,1,0,0,0] Flip nums[5],nums[6],nums[7]: nums becomes

[1,1,1,1,1,1,1,1]

Constraints:

* `1 <= nums.length <= 105` * `1 <= k <= nums.length`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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