2134. Minimum Swaps to Group All 1's Together II

A **swap** is defined as taking two **distinct** positions in an array and swapping the values in them.

A **circular** array is defined as an array where we consider the **first** element and the **last** element to be **adjacent**.

Given a **binary circular** array `nums`, return *the minimum number of swaps required to group all* `1`*'s present in the array together at* ***any location***.

Example 1:

```
Input: nums = [0,1,0,1,1,0,0]
Output: 1
Explanation: Here are a few of the ways to group all the 1's together:
[0,0,1,1,1,0,0] using 1 swap.
[0,1,1,1,0,0,0] using 1 swap.
[1,1,0,0,0,0,1] using 2 swaps (using the circular property of the array).
There is no way to group all 1's together with 0 swaps.
Thus, the minimum number of swaps required is 1.
```

Example 2:

```
Input: nums = [0,1,1,1,0,0,1,1,0]
Output: 2
Explanation: Here are a few of the ways to group all the 1's together:
[1,1,1,0,0,0,0,1,1] using 2 swaps (using the circular property of the array).
[1,1,1,1,1,0,0,0,0] using 2 swaps.
There is no way to group all 1's together with 0 or 1 swaps.
Thus, the minimum number of swaps required is 2.
```

Example 3:

```
Input: nums = [1,1,0,0,1]
Output: 0
Explanation: All the 1's are already grouped together due to the circular property
of the array.
Thus, the minimum number of swaps required is 0.
```

Constraints:

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

2134. Minimum Swaps to Group All 1's Together II

```
/*
    Prercessing+Sliding window
    Time complexity: O(5n)=O(n)
    Space complexity: O(2n)=O(n)
*/
typedef std::vector<int> vi;
class Solution {
public:
    int minSwaps(vector<int>& nums) {
        int nb_ones=0;
        for(auto& e: nums) if(e==1) nb_ones++;

        int n=nums.size();
        for(int i=0;i<n;++i) nums.push_back(nums[i]);

        n=2*n;

        vi pre_ones(n+1,0),pre_zeros(n+1,0);

        for(int i=0;i<n;++i){
            pre_ones[i+1]=nums[i]==1?pre_ones[i]+1:pre_ones[i];
        }

        int i=0,j=nb_ones-1,max_ones_in_win=INT_MIN;
        while(j<n){
            int cnt_ones_in_win=pre_ones[j+1]-pre_ones[i];

           max_ones_in_win=std::max(max_ones_in_win,cnt_ones_in_win);
            i++;
            j++;
        }

        return nb_ones-max_ones_in_win;
    }
};
```

2134. Minimum Swaps to Group All 1's Together II

```
/*
    Sliding window
    Time complexity: O(4n)=O(n)
    Space complexity: O(n)
*/
typedef std::vector<int> vi;
class Solution {
public:
    int minSwaps(vector<int>& nums) {
        // Count total number of ones
        int nb_ones=0;
        for(auto& e: nums) if(e==1) nb_ones++;

        // Extend the array for circularity
        int n=nums.size();
        for(int i=0;i<n;++i) nums.push_back(nums[i]);

        n=2*n;

        //Create a window of size nb_ones
        int cnt_ones_in_win=0;
        for(int i=0;i<nb_ones;++i){
            if (nums[i]==1) cnt_ones_in_win++;
        }

        // Slide the window
        int i=0,max_ones_in_win=INT_MIN;
        for(int i=0;i<n-nb_ones;++i){

           max_ones_in_win=std::max(max_ones_in_win,cnt_ones_in_win);
            cnt_ones_in_win-=nums[i];
            cnt_ones_in_win+=nums[i+nb_ones];
        }

        return nb_ones-max_ones_in_win;
    }
};
```