Assignment Solutions | Sliding window | Week 12

1. Given an array of integers `arr` and two integers `k` and `threshold`, return *the number of sub-arrays of size* `k` *and average greater than or equal to* `threshold`. [Leetcode 1343]

   Example 1:

   Input: arr = [2,2,2,2,5,5,5,8], k = 3, threshold = 4

   Output: 3

   Explanation: Sub-arrays [2,5,5],[5,5,5] and [5,5,8] have averages 4, 5 and 6 respectively. All other sub-arrays of size 3 have averages less than 4 (the threshold).

   Example 2:

   Input: arr = [11,13,17,23,29,31,7,5,2,3], k = 3, threshold = 5

   Output: 6

   Explanation: The first 6 sub-arrays of size 3 have averages greater than 5. Note that averages are not integers.

Solution :

```cpp
class Solution {

public:
    int numOfSubarrays(vector<int>& a, int k, int th) {
        int n = a.size();

        int sum = 0;
        int avg = 0;
        int ans = 0;
        for(int i=0;i<k;i++)sum += a[i];
        avg = sum/k;
        if(avg >= th)ans++;

        int i=k;

        while(i < n){
            sum -= a[i-k];
            sum += a[i];
            avg = sum/k;
            if(avg >= th)ans++;
            i++;
        }
```

2. The **score** of an array is defined as the **product** of its sum and its length.

- For example, the score of `[1, 2, 3, 4, 5]` is `(1 + 2 + 3 + 4 + 5) * 5 = 75`.

  Given a positive integer array `nums` and an integer `k`, return *the **number of non-empty subarrays** of* `nums` *whose score is **strictly less** than* `k`.

  A **subarray** is a contiguous sequence of elements within an array. [Leetcode 2302]

  Example 1:

  Input: nums = [2,1,4,3,5], k = 10

  Output: 6

  Explanation:

  The 6 subarrays having scores less than 10 are:

  - [2] with score 2 * 1 = 2.
  - [1] with score 1 * 1 = 1.
  - [4] with score 4 * 1 = 4.
  - [3] with score 3 * 1 = 3.
  - [5] with score 5 * 1 = 5.
  - [2,1] with score (2 + 1) * 2 = 6.

  Note that subarrays such as [1,4] and [4,3,5] are not considered because their scores are 10 and 36 respectively, while we need scores strictly less than 10.

Example 2:

Input: nums = [1,1,1], k = 5

Output: 5

Explanation:

Every subarray except [1,1,1] has a score less than 5.

[1,1,1] has a score (1 + 1 + 1) * 3 = 9, which is greater than 5.

Thus, there are 5 subarrays having scores less than 5.

Solution :

```
class Solution {
public:
    long long countSubarrays(vector<int>& a, long long k) {
        long long int i=0,j=0,sum=0,score=0,ans=0;

        long long int n = a.size();
        while(i < n and j < n){
            sum += a[j]; //window expansion
            score = sum*(j-i+1);

            while(i<=j and score >= k){
                //window contraction
                sum -= a[i++];
                score = sum*(j-i+1);
            }

            ans += (j-i+1);
            j++;
        }
        return ans;
    }
};
```

3. Given an array of integers `nums` and an integer `k`. A continuous subarray is called **nice** if there are `k` odd numbers on it. [Leetcode 1248]

   Return *the number of **nice** sub-arrays*.

   Example 1:

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

   Output: 2

   Explanation: The only sub-arrays with 3 odd numbers are [1,1,2,1] and [1,2,1,1].

   Example 2:

   Input: nums = [2,4,6], k = 1

   Output: 0

   Explanation: There is no odd numbers in the array.

   Example 3:

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

Output: 16

Solution :

```
class Solution {
public:
    int numberOfSubarrays(vector<int>& a, int k) {
        int n = a.size();
        int i=0,j=0,cnt=0,ans=0,odd=0;

        while(j<n){
            if(a[j]%2!=0){
                cnt = 0;
                odd++;
            }
            while(i<=j and odd == k){
                cnt++;
                if(a[i++]%2 != 0)odd--;
            }
            ans += cnt;
            j++;
        }
        return ans;
    }
};
```
