First try fixed size window with following questions:

Steps:

1) find the sum of all elements in an array

2) find the max sum of subarray with size k

**Recommended Fixed-Size Practice Sequence**

**Step 1 — Pure sums**

1. **LC 643 — Max Average Subarray I**
   * Sum over k elements, track max.
2. **LC 1343 — Number of Subarrays of Size K and Avg ≥ Threshold**
   * Same as above, but check condition instead of max.

**Step 2 — Counting inside window**

1. **LC 1456 — Maximum Number of Vowels in a Substring of Given Length**
   * Keep a vowel count as you slide.
2. **LC 1208 — Get Equal Substrings Within Budget** *(can fix k and solve)*
   * Maintain cost sum for exactly k characters. (variable sliding window)

**Step 3 — Max/min value inside window**

1. **LC 239 — Sliding Window Maximum** *(hard, but fixed size)*
   * Use deque to get max in O(1) per slide. (Its actually a Queue problem)

**Step 4 — Codeforces fixed-size**

1. **CF 279B — Books** *(adapt to fixed-size version by constraining k)*
2. **CF 1690C — Restoring the Duration of Tasks** *(window simulation)*

**Easiest LeetCode Variable Sliding Window Problems**

Here’s a gentle progression — all can be solved in **O(n)** with two pointers:

1. **LC 3 — Longest Substring Without Repeating Characters**
   * Condition: all characters in the window must be unique.
   * Adjust left pointer when a duplicate is found.
2. **LC 904 — Fruit Into Baskets**
   * Condition: at most 2 types of fruit in the basket.
   * Adjust left pointer when there are more than 2 types.
3. **LC 209 — Minimum Size Subarray Sum**
   * Condition: sum of window ≥ target.
   * Adjust left pointer while condition is satisfied to shrink window.
4. **LC 1004 — Max Consecutive Ones III**
   * Condition: can flip at most k zeros to ones.
   * Adjust left pointer when zero count > k.
5. **LC 1208 — Get Equal Substrings Within Budget**
   * Condition: total “cost” ≤ budget.
   * Adjust left pointer when cost exceeds budget.