Sliding Window Pattern: Explained

1. Definition

The **Sliding Window** is a technique used to solve problems that involve **contiguous segments or subarrays** of a given array or string.

- It involves maintaining a window (a range or segment) defined by two pointers (usually start and end) that slide over the input data.
- The window can either be fixed size or variable size.
- Instead of recalculating everything for every window, it efficiently updates results as the window moves forward by adding the new element and removing the old element.

2. Where it can apply?

The sliding window pattern applies mainly when:

- You need to find something about **subarrays or substrings** of contiguous elements.
- You want to calculate or track a property over a range that moves forward step-by-step.
- Problems involving sums, counts, maximum/minimum, frequency of elements inside a window.
- Fixed-size windows (e.g., "find max sum of subarray of size k")
- Variable-size windows where the window expands or shrinks based on some condition (e.g., "smallest subarray with sum >= target")

3. How to apply the Sliding Window pattern?

There are generally two types of sliding windows:

a) Fixed-Size Sliding Window

- Window size is fixed (say k).
- Move the window from the start to the end by sliding one element at a time.
- For each move, update the result by adding the new element and removing the element that slides out.

Example steps:

- Initialize sum, frequency map, or required tracking variable for the first k elements.
- Slide the window by one element forward:
  - Remove the leftmost element from the window.
  - Add the new rightmost element.
- Update your result or answer for the current window.

b) Variable-Size Sliding Window

- Window size changes dynamically based on a condition.
- Use two pointers: start and end.
- Move end forward to expand the window until a condition is met.
- Move start forward to shrink the window to satisfy constraints or optimize the solution.

Example steps:

- Initialize start and end to 0.
- Expand end to include new elements and update your tracking variables.
- When the condition is met (e.g., window sum >= target), try shrinking the window by moving start forward while maintaining the condition.
- Keep track of the best (minimum, maximum, count, etc.) window found so far.

4. How to check which type of questions sliding window can apply to?

Look for these clues in the problem:

- The problem deals with **subarrays or substrings**.
- You need to find something like:
  - Maximum/minimum sum of a subarray of size k.
  - Longest/shortest substring/subarray meeting some criteria.
  - Count of distinct elements or frequencies in a substring/subarray.
  - Any problem involving "continuous" or "contiguous" segments.
- The problem hints at "window", "substring", "subarray", or "contiguous".
- Naive solution would involve nested loops iterating over all subarrays; sliding window offers optimization.
- Constraints are large (like 10^5 elements) where brute force O(n^2) is not feasible.

5. Benefits of Sliding Window

- **Efficiency:** Reduces time complexity from $O(n^2)$ (nested loops) to O(n).
- **Simplicity:** Easy to implement once understood.
- **Memory:** Uses constant or linear extra space, depending on implementation.
- **Versatility:** Works for fixed-size and variable-size problems.
- Ideal for real-time streaming data and online algorithms where data is processed sequentially.

No.	Problem Name	Difficulty	Platform	Link
✓ 1	Maximum Average Subarray I	Easy	LeetCode	Link
2	Minimum Size Subarray Sum	Medium	LeetCode	Link
3	Longest Substring Without Repeating Characters	Medium	LeetCode	Link

✓ 4	Permutation in String	Medium	LeetCode	Link
✓ 5	Find All Anagrams in a String	Medium	LeetCode	Link
✓ 6	Sliding Window Maximum	Hard	LeetCode	Link
7	Longest Repeating Character Replace-ment	Medium LeetCode Link		
8	Count Number of Nice Subarrays	Medium	LeetCode	Link
9	Longest Substring with At Most K Distinct Characters	Medium	LeetCode	Link
10	Substring with Concatena-tion of All Words	Hard LeetCode Link		
11	Maximum Number of Vowels in a Substring of	Medium	LeetCode	Link

	Given Length			
12	Count Good Substrings	Easy	LeetCode	Link
13	Longest Substring Without 3 Identical Consecutive Characters	Medium	LeetCode	Link
14	Maximum Erasure Value	Medium	LeetCode	Link
15	Max Consecutive Ones III	Medium	LeetCode	Link