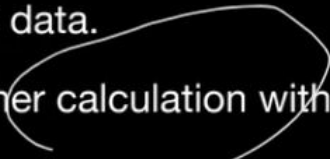# Sliding Window

- The sliding window technique is a common approach used to solve problems that involve finding or analyzing a subset of elements in a larger data structure, such as an array or string. Here are the key patterns often used in sliding window problem statements:

  - Fixed Size Window

  - Variable Size Window

  - Two Pointer Approach

  - Optimised Sliding Windows

Fixed-Size Windows

- Fixed Size Window:

  - Maintain a window of a fixed size (usually a contiguous subset of elements).

  - Slide the window one element at a time to process the data.

  - Useful for finding the maximum, minimum, or some other calculation within a fixed-sized window.

Problem Statements:

- Fixed Size Window:

  - Given an array of integers and a window size, find the maximum sum of any subarray of the given window size.

  - Given a string, find the length of the longest substring containing only unique characters within a fixed window size.

  - Given an array of temperatures, find the minimum number of days it takes for a contiguous window of size K to have increasing temperatures.

  - Given a binary array (containing only 0s and 1s), find the maximum number of consecutive 1s with at most K 0s allowed in the window.

  - Given a string of characters, find the smallest window length that contains all the characters of a given pattern.

Variable-Size Window

- Variable Size Window:

  - Start with a small window and gradually expand or shrink it based on certain conditions.

  - Useful when the size of the window is not fixed and depends on the problem requirements.

  - Often used to find a subarray or substring satisfying specific conditions (e.g., sum, product, or a certain property).

Problem Statements:

- Variable Size Window:

  - Given an array of positive integers and a target sum, find the minimum subarray length whose sum is greater than or equal to the target.

  - Given a string, find the longest substring with at most K distinct characters.

  - Given an array of non-negative integers, find the subarray of length K with the maximum average value.

  - Given a string, find the length of the longest substring that can be formed by repeating exactly K times.

  - Given an array of integers, find the maximum product of a subarray with size at most K.

2-Pointer Approach

- Two Pointers:

  - Use two pointers to define the boundaries of the window.

  - One pointer usually represents the start of the window, and the other represents the end.

  - Move the pointers within the data structure based on certain conditions.

  - Commonly used when the problem requires finding a target sum, a pair of elements, or a subarray with specific characteristics.

Problem Statements:

- Two Pointers:

  - Given an array of integers and a target sum, find two numbers that add up to the target.

  - Given a sorted array of integers, find two numbers whose absolute difference is closest to a given target value.

  - Given a string, find the longest palindrome substring.

  - Given a string, find the length of the longest substring without repeating characters.

  - Given an array of integers, find a subarray with the given sum.

Optimised Window:

- Optimized Sliding Window:

  - Apply additional optimization techniques to improve the sliding window solution.

  - Examples include precomputing sums or frequencies, utilizing data structures like heaps or hash maps, or skipping unnecessary calculations.

  - Helps to reduce the time complexity and optimize the overall solution.

Problem Statements:

- Optimized Sliding Window:

  - Given an array of integers, find the subarray with the maximum sum using the Kadane's algorithm.

  - Given a string, find the longest substring with at most K distinct characters using a hash map for character frequency tracking.

  - Given an array of integers, find the median of every sliding window of size K using a min heap and max heap.

  - Given a binary array (containing only 0s and 1s), find the minimum size of a subarray that needs to be flipped to make all elements 1.

  - Given an array of integers and a target sum, find the subarray with the smallest length that sums up to the target using a two-pointer approach and dynamic window resizing.

Problem Statements:

- Given a string, find the longest substring that contains at most K distinct characters and has at least K/2 repeating characters.

- Given an array of integers, find the maximum product of a subarray with size at most K and the minimum product of a subarray with size at most K.

- Given two strings, find the minimum window in the first string that contains all the characters of the second string, considering multiple possible windows.

- Given a string, find the longest palindromic substring with at most K deletions allowed.

- Given an array of integers, find the maximum sum of K non-overlapping subarrays.