What is the Two Pointers Pattern?

The **Two Pointers** technique involves using two indices (pointers) to iterate over a data structure (usually an array or a string) to solve problems efficiently by avoiding nested loops.

When to Use Two Pointers?

- When you need to find pairs, triplets, or subarrays meeting certain conditions.
- When the data is sorted or can be sorted.
- When you want to optimize brute force solutions that use nested loops ($O(n^2)$) to linear or near-linear time ($O(n)$).

How It Works?

You maintain two pointers that move through the data structure according to certain rules:

- **One pointer starts at the beginning**, the other at the **end** (common in problems like finding pairs with a sum).
- Or, **both pointers start at the beginning**, with one moving faster than the other (useful for sliding window problems).
- Move pointers towards each other or forward depending on the problem condition.

Typical Approach:

1. Initialize two pointers, left and right.
2. Check condition based on the current pointers.
3. Move pointers accordingly:
    - If condition not met, move left or right pointer to try to satisfy the condition.
    - If condition met, record the answer or move pointers to find more solutions.
4. Repeat until pointers cross or reach the end.

#	Problem Name	Platform Link
1	Remove Duplicates from Sorted Array	LeetCode 26
2	Two Sum II - Input Array Is Sorted	LeetCode 167
3	Move Zeroes	LeetCode 283
4	Reverse String	LeetCode 344
5	Container With Most Water	LeetCode 11
6	Valid Palindrome	LeetCode 125
7	Squares of a Sorted	LeetCode 977

	Array	
8	Subarray Product Less Than K	LeetCode 713
9	Remove Element	LeetCode 27
10	3Sum	LeetCode 15
11	Sort Colors (Dutch National Flag Problem)	LeetCode 75
12	Longest Substring Without Repeating Characters	LeetCode 3
13	Minimum Size Subarray Sum	LeetCode 209
14	Trapping Rain Water	LeetCode 42
15	Longest Mountain in Array	LeetCode 845