7 Techniques to solve most Array problems

➤ Two-Pointer Technique:

-> This involves using two pointers to traverse the array. It is particularly useful for problems involving pairs, such as finding two numbers that sum up to a target, or reversing an array.

-> Example: Finding if there are two elements that add up to a given sum.

1. Two Sum II - Input Array Is Sorted: https://lnkd.in/d69eyq-k

2. 3Sum: <https://lnkd.in/dd6mwshC>

3. Container With Most Water: <https://lnkd.in/derBVWA2>

➤ Hashing:

-> Using a hash table (or dictionary) can help with quick lookups and is useful for problems involving frequency counts or checking for the presence of elements.

-> Example: Finding the first repeating element or checking for subarray with a sum of zero.

1. Subarray Sum Equals K: https://lnkd.in/dfVQuvEZ

2. Two Sum: https://lnkd.in/dWDJFCrk

3. Longest Consecutive Sequence: https://lnkd.in/d9QdkcUN

➤ Sorting:

-> Sorting the array first can simplify many problems, especially those involving ordering, searching for pairs, or intervals.

-> Examples:

1. 3Sum: https://lnkd.in/dd6mwshC

2: Merge Intervals: https://lnkd.in/dBwpCyfT

3. Sort Colors: https://lnkd.in/deTf9AGz

➤ Sliding Window:

-> This technique is used for problems involving subarrays or contiguous segments of an array. It helps in finding the maximum or minimum sum of a subarray of a given size.

-> Example: Finding the maximum sum subarray of size k.

1. Maximum Average Subarray I: https://lnkd.in/dY\_Kwxdf

2. Longest Substring Without Repeating Characters: https://lnkd.in/dMYzuAY4

3. Minimum Size Subarray Sum: https://lnkd.in/dNHBmWyW

➤ Dynamic Programming:

-> This technique involves solving problems by breaking them down into simpler subproblems and storing the results to avoid redundant computations. It is useful for problems like finding the longest increasing subsequence or maximum subarray sum.

-> Example: Finding the maximum sum subarray Kadane algorithm.

1. Maximum Subarray](https://lnkd.in/dA3e79e7)

2. Longest Increasing Subsequence: https://lnkd.in/dSkJXnDp

3. House Robber: https://lnkd.in/dtY5BgDD

➤ Greedy Algorithms:

-> Greedy algorithms make a sequence of choices, each of which looks the best at the moment, with the hope of finding the global optimum. This approach works well for optimization problems.

-> Example:

1. Finding the minimum number of platforms needed for a railway station.

2. Meeting Rooms II: https://lnkd.in/dN\_NRPWg

3. Jump Game: https://lnkd.in/ddHjkVbb

➤ Divide and Conquer:

-> This technique involves dividing the problem into smaller subproblems, solving them independently, and then combining the results. It is often used in sorting algorithms like merge sort and quick sort.

-> Example:

1. Find Peak Element: https://lnkd.in/dBuHGUBZ

2. Merge Sort: https://lnkd.in/d5TVqNCE (Implement Merge Sort)

3. Kth Largest Element in an Array: https://lnkd.in/dE8\_7r4m

Stay curious, keep learning, keep sharing !