

Capgemini Coding Problems Collection –

By Durgesh StudyHub

Capgemini Coding Problems Collection – Part 1

Description: This document contains **50 curated Capgemini-style coding problems** with **problem statements, sample input & output, and detailed solution explanations** (intuition and steps), **no code**. This is *Part 1* of a planned 500+ problem collection. If you'd like, I will continue and add more parts until we reach 500+.

Problem 1 — Two Sum (Array)

Problem: Given an array of integers and a target sum, find two numbers that add up to the target and return their indices (1-based). If multiple answers, return any one pair.

Sample Input: arr = [2, 7, 11, 15], target = 9 **Sample Output:** 1 2

Explanation / Approach:

- Use a hash map to store value → index as you scan the array.
- For each element x , check if $\text{target} - x$ exists in the map.
- If yes, we've found a pair; return indices.

Complexity: O(n) time, O(n) space.

Problem 2 — Reverse Words in a String

Problem: Given a string containing spaces, reverse the order of words. Words are separated by one or more spaces. Trim leading/trailing spaces and reduce multiple spaces to a single space.

Sample Input: " the sky is blue " **Sample Output:** "blue is sky the"

Explanation / Approach:

- Split the string by whitespace into words.
- Reverse the list of words and join with single spaces.
- Alternatively, use two-pointer scanning to extract words and build result.

Complexity: O(n) time, O(n) extra space.

Problem 3 — Merge Intervals

Problem: Given a list of intervals, merge all overlapping intervals and return the merged list sorted by start time.

Sample Input: `[[1,3],[2,6],[8,10],[15,18]]` **Sample Output:** `[[1,6],[8,10],[15,18]]`

Explanation / Approach:

- Sort intervals by start.
- Iterate and keep a current interval; if the next overlaps (`next.start <= current.end`), merge by extending end.
- Otherwise, append current and move to next.

Complexity: O(n log n) time due to sort, O(n) space.

Problem 4 — Valid Parentheses

Problem: Given a string containing just the characters `()[]{}` determine if the input string is valid (properly closed and nested).

Sample Input: `"()[]{}"` **Sample Output:** `true`

Explanation / Approach:

- Use a stack. Push opening brackets. For closing bracket, check top of stack for matching opening.
- If mismatch or stack empty when closing appears → invalid. At end, stack must be empty.

Complexity: O(n) time, O(n) space.

Problem 5 — Maximum Subarray (Kadane)

Problem: Given an integer array, find the contiguous subarray with the largest sum and return that sum.

Sample Input: `[-2,1,-3,4,-1,2,1,-5,4]` **Sample Output:** 6 (subarray `[4,-1,2,1]`)

Explanation / Approach:

- Kadane's algorithm: maintain $\text{currentMax} = \max(\text{element}, \text{currentMax} + \text{element})$, and overall max.
- Handles all-negative arrays by taking max element.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 6 — Fibonacci (nth number) with memo

Problem: Compute nth Fibonacci number (0-indexed or 1-indexed as specified) efficiently.

Sample Input: $n = 10$ (0-indexed means $F_0 = 0, F_1 = 1$) **Sample Output:** 55

Explanation / Approach:

- Use DP or memoization to avoid exponential recursion.
- Iterative bottom-up requires $O(n)$ time and $O(1)$ space if only last two stored.

Complexity: $O(n)$ time, $O(1)$ space (iterative).

Problem 7 — Climbing Stairs (DP)

Problem: You can climb 1 or 2 steps. How many distinct ways to reach top of n steps?

Sample Input: $n = 3$ **Sample Output:** 3

Explanation / Approach:

- $\text{Ways}(n) = \text{Ways}(n-1) + \text{Ways}(n-2)$.
- Base: $\text{Ways}(1)=1, \text{Ways}(2)=2$. Compute iteratively.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 8 — Best Time to Buy and Sell Stock (Single transaction)

Problem: Given prices where $\text{prices}[i]$ is the price on day i , max profit from one buy-sell.

Sample Input: [7, 1, 5, 3, 6, 4] **Sample Output:** 5 (buy at 1, sell at 6)

Explanation / Approach:

- Track `minPrice` so far and compute `profit = price - minPrice` at each step; keep max.

Complexity: O(n) time, O(1) space.

Problem 9 — Product of Array Except Self

Problem: Given `nums`, return array `output` where `output[i]` is product of all elements except `nums[i]`. Do it without division and in O(n).

Sample Input: [1, 2, 3, 4] **Sample Output:** [24, 12, 8, 6]

Explanation / Approach:

- Build left-products and right-products (prefix and suffix), multiply per index.
- Use two passes and O(1) extra space (excluding output) by accumulating right product on the fly.

Complexity: O(n) time, O(1) extra space.

Problem 10 — Search in Rotated Sorted Array

Problem: A sorted array is rotated. Given target, return its index or -1.

Sample Input: `nums=[4, 5, 6, 7, 0, 1, 2]`, `target=0` **Sample Output:** 4

Explanation / Approach:

- Modified binary search. Detect which half is sorted via comparisons and decide where to search next.

Complexity: O(log n) time.

Problem 11 — Majority Element

Problem: Given array, find element appearing more than n/2 times.

Sample Input: [3, 2, 3] **Sample Output:** 3

Explanation / Approach:

- Boyer-Moore Voting Algorithm: keep candidate and count. Two passes (optional) to verify.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 12 — Container With Most Water

Problem: Given heights, find two lines that together with x-axis form container with max area.

Sample Input: [1, 8, 6, 2, 5, 4, 8, 3, 7] **Sample Output:** 49

Explanation / Approach:

- Two-pointer approach: start at ends, move shorter pointer inward, track max area.
- Intuition: moving the taller pointer cannot help increase area if shorter fixed.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 13 — 3Sum

Problem: Find unique triplets that sum to zero.

Sample Input: [-1, 0, 1, 2, -1, -4] **Sample Output:** [[[-1, -1, 2], [-1, 0, 1]]]

Explanation / Approach:

- Sort array. Fix one element, then run two-pointer two-sum for remaining part, skip duplicates.

Complexity: $O(n^2)$ time.

Problem 14 — Longest Palindromic Substring

Problem: Find the longest palindromic substring in a given string.

Sample Input: "babad" **Sample Output:** "bab" (or "aba")

Explanation / Approach:

- Expand-around-center for each index (odd and even centers) $\rightarrow O(n^2)$.

- Manacher's algorithm achieves $O(n)$ but is complex; expansion is acceptable for interview.

Complexity: $O(n^2)$ time, $O(1)$ space.

Problem 15 — Valid Anagram

Problem: Given two strings, check if one is an anagram of the other.

Sample Input: `s = "anagram", t = "nagaram"` **Sample Output:** `true`

Explanation / Approach:

- Count frequency of characters in both strings (hash map or fixed 26-length array). Compare counts.

Complexity: $O(n)$ time, $O(1)$ extra space (alphabet-limited).

Problem 16 — Group Anagrams

Problem: Given array of strings, group anagrams together.

Sample Input: `["eat", "tea", "tan", "ate", "nat", "bat"]` **Sample Output:** `[["eat", "tea", "ate"], ["tan", "nat"], ["bat"]]`

Explanation / Approach:

- Use sorted string or character count as key in hash map to group.

Complexity: $O(n * k \log k)$ if sorting each string, or $O(n * k)$ using counts (k = string length).

Problem 17 — Pow(x, n)

Problem: Implement `pow(x, n)` — compute x^n (n may be negative).

Sample Input: `x=2.00000, n=10` **Sample Output:** `1024.00000`

Explanation / Approach:

- Fast exponentiation: divide-and-conquer exponentiation (exponent halving).
- Handle negative n via reciprocal.

Complexity: $O(\log n)$ time.

Problem 18 – Counting Bits

Problem: For each number $0..n$, compute the number of 1 bits in binary representation.

Sample Input: $n=5$ **Sample Output:** $[0, 1, 1, 2, 1, 2]$

Explanation / Approach:

- Use DP relation: $\text{bits}[i] = \text{bits}[i \& (i-1)] + 1$, or $\text{bits}[i] = \text{bits}[i \gg 1] + (i \& 1)$.

Complexity: $O(n)$ time.

Problem 19 – Single Number (XOR)

Problem: Every element appears twice except one. Find that single one.

Sample Input: $[2, 2, 1]$ **Sample Output:** 1

Explanation / Approach:

- XOR all numbers: duplicates cancel, leaving unique.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 20 – Linked List Cycle (detect)

Problem: Given a linked list, determine if it has a cycle.

Sample Input: head with cycle at node 2 **Sample Output:** true

Explanation / Approach:

- Floyd's tortoise and hare: slow and fast pointers. If they meet, cycle exists.

Complexity: $O(n)$ time, $O(1)$ space.

Problem 21 — Remove Duplicates from Sorted Array

Problem: Given sorted array, remove duplicates in-place and return new length.

Sample Input: [1, 1, 2] **Sample Output:** 2 (array becomes [1, 2])

Explanation / Approach:

- Two pointers: one for iteration, one for position to write unique values.

Complexity: O(n) time, O(1) space.

Problem 22 — Rotate Image (Matrix)

Problem: Rotate an n x n matrix by 90 degrees clockwise in-place.

Sample Input: [[1, 2, 3], [4, 5, 6], [7, 8, 9]] **Sample Output:** [[7, 4, 1], [8, 5, 2], [9, 6, 3]]

Explanation / Approach:

- Transpose the matrix, then reverse each row. Both operations in-place.

Complexity: O(n^2) time, O(1) extra space.

Problem 23 — Spiral Matrix

Problem: Given m x n matrix, return elements in spiral order.

Sample Input: [[1, 2, 3], [4, 5, 6], [7, 8, 9]] **Sample Output:** [1, 2, 3, 6, 9, 8, 7, 4, 5]

Explanation / Approach:

- Maintain four boundaries (top, bottom, left, right) and iterate while adjusting boundaries.

Complexity: O(mn) time.

Problem 24 – Minimum Path Sum (Grid)

Problem: Given grid of non-negative numbers, find path from top-left to bottom-right that minimizes sum (only right or down moves).

Sample Input: `[[1, 3, 1], [1, 5, 1], [4, 2, 1]]` **Sample Output:** 7 (1→3→1→1→1)

Explanation / Approach:

- DP: $dp[i][j] = grid[i][j] + \min(dp[i-1][j], dp[i][j-1])$. Can do in-place to save space.

Complexity: O(mn) time, O(n) space with optimization.

26. Missing Number Finder

Problem: Given n numbers from 1 to $n+1$ with one number missing, find the missing number.

Input: [1, 2, 4, 5, 6]

Output: 3

Explanation: The sum of numbers 1–6 is 21, actual sum is 18, missing = 21–18 = 3.

27. Palindrome String Check

Problem: Check whether the given string is palindrome.

Input: madam

Output: Yes

Explanation: Reading from both ends gives the same string.

28. Second Largest Element

Problem: Find the second largest number in an array.

Input: [12, 35, 1, 10, 34, 1]

Output: 34

Explanation: Largest is 35, second largest is 34.

29. Count Vowels in a String

Problem: Count vowels in the input string.

Input: education

Output: 5

Explanation: Vowels are e, u, a, i, o.

30. Anagram Checker

Problem: Check if two strings are anagrams.

Input: listen, silent

Output: Yes

Explanation: Both strings have same letters with same frequency.

31. Prime Range Counter

Problem: Count primes between two numbers.

Input: 10 30

Output: 6

Explanation: Primes = 11,13,17,19,23,29.

32. Reverse a Sentence

Problem: Reverse the order of words in a sentence.

Input: “I love Capgemini”

Output: “Capgemini love I”

Explanation: Reverse the word order.

33. Armstrong Number

Problem: Check if number is Armstrong.

Input: 153

Output: Yes

Explanation: $1^3 + 5^3 + 3^3 = 153$.

34. Perfect Number Check

Problem: Check if number equals sum of its divisors.

Input: 28

Output: Yes

Explanation: $1+2+4+7+14=28$.

35. Fibonacci Series Generator

Problem: Print first N Fibonacci numbers.

Input: 7

Output: 0 1 1 2 3 5 8

Explanation: Each term is sum of previous two.

36. Factorial Finder

Problem: Compute factorial of a number.

Input: 5

Output: 120

Explanation: $5 \times 4 \times 3 \times 2 \times 1 = 120$.

37. Remove Duplicates from Array

Problem: Remove duplicate elements while maintaining order.

Input: [1,2,2,3,4,4,5]

Output: [1,2,3,4,5]

Explanation: Only unique elements kept.

38. Sum of Digits

Problem: Calculate sum of digits of number.

Input: 1234

Output: 10

Explanation: $1+2+3+4=10$.

39. String Compression

Problem: Compress repeated characters in string.

Input: aaabbccc

Output: a3b2c3

Explanation: Consecutive counts replaced by number.

40. Count Words in Sentence

Problem: Count number of words in sentence.

Input: “Capgemini is great”

Output: 3

Explanation: Three space-separated words.

41. Binary to Decimal

Problem: Convert binary to decimal.

Input: 1010

Output: 10

Explanation: $(1 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1) = 10$.

42. Decimal to Binary

Problem: Convert decimal number to binary.

Input: 13

Output: 1101

Explanation: Divide by 2 repeatedly.

43. Find GCD (Greatest Common Divisor)

Problem: Compute GCD of two numbers.

Input: 12 18

Output: 6

Explanation: Common divisors: 1,2,3,6.

44. LCM of Two Numbers

Problem: Compute Least Common Multiple.

Input: 4 6

Output: 12

Explanation: 12 is smallest divisible by 4 and 6.

45. Check Substring Presence

Problem: Check if one string contains another.

Input: “Capgemini India” and “India”

Output: Yes

Explanation: India appears in main string.

46. Remove Spaces from String

Problem: Remove all spaces in input.

Input: "I love coding"

Output: "Ilovecoding"

47. Power of Number

Problem: Compute a^b .

Input: 2 5

Output: 32

Explanation: $2^5 = 32$.

48. Find Median of Array

Problem: Find median value of numbers.

Input: [1,3,5,2,4]

Output: 3

Explanation: Sorted = [1,2,3,4,5]. Middle = 3.

49. Rotate Array by K

Problem: Rotate array elements right by K.

Input: [1,2,3,4,5], K=2

Output: [4,5,1,2,3]

50. Sum of Even Numbers

Problem: Sum all even numbers in array.

Input: [1,2,3,4,5,6]

Output: 12

Explanation: $2+4+6=12$.

51. Replace Character in String

Problem: Replace specific character with another.

Input: “banana”, replace a→o

Output: “bonono”

52. Leap Year Check

Problem: Determine if year is leap year.

Input: 2024

Output: Yes

Explanation: Divisible by 4 and not 100 unless 400.

53. Matrix Diagonal Sum

Problem: Find sum of primary diagonal in $N \times N$ matrix.

Input: [[1,2,3],[4,5,6],[7,8,9]]

Output: 15

Explanation: $1+5+9=15$.

54. Find Maximum Occurring Character

Problem: Find character with highest frequency.

Input: "mississippi"

Output: i

Explanation: i appears 4 times.

55. Palindrome Number Check

Problem: Check if number is palindrome.

Input: 121

Output: Yes

Explanation: Same when reversed.

56. Find Missing Character in Sequence

Problem: Given sequence of alphabets missing one, find it.

Input: a,b,c,e

Output: d

Explanation: ASCII difference detects gap.

57. Sum of Natural Numbers

Problem: Sum of first N natural numbers.

Input: 10

Output: 55

Explanation: $n(n+1)/2 = 55$.

58. Count Digits in Number

Problem: Count number of digits.

Input: 98765

Output: 5

Explanation: Number length = 5.

59. Pattern Printing (Triangle)

Problem: Print right-angled triangle of * for N.

Input: 4

Output: * **

Explanation: Increasing stars per line.

60. Count Occurrences of Element

Problem: Count how many times a number appears in array.

Input: [1,2,2,3,2,4], element=2

Output: 3

Explanation: 2 appears 3 times.

61. Merge Two Sorted Arrays

Problem: Merge two sorted arrays into one sorted array.

Input: [1,3,5], [2,4,6]

Output: [1,2,3,4,5,6]

Explanation: Use two pointers to merge in linear time.

62. Check Power of Two

Problem: Determine if a number is power of two.

Input: 16

Output: Yes

Explanation: Only one bit set in binary representation.

63. Sum of Two Largest Numbers

Problem: Find sum of two largest numbers in array.

Input: [5,1,9,7]

Output: 16

Explanation: $9 + 7 = 16$.

64. Remove Given Element

Problem: Remove all occurrences of a value in-place and return new length.

Input: [3,2,2,3], val=3

Output: 2 (array becomes [2,2])

Explanation: Overwrite unwanted values with two-pointer technique.

65. Check Anagram Palindrome

Problem: Determine if any permutation of string can form a palindrome.

Input: "carrace"

Output: Yes

Explanation: Can be rearranged to "racecar".

66. Sum of Prime Factors

Problem: Sum distinct prime factors of a number.

Input: 28

Output: 10

Explanation: Prime factors 2 and 7 → $2+7=9$ (if counting multiplicity add accordingly).

(Clarify whether multiplicity counted — here distinct assumed.)

67. Validate Email Format

Problem: Check if input string is valid email format (basic checks).

Input: "user@example.com"

Output: Yes

Explanation: Contains one '@', domain with dot, and valid local part characters.

68. Sum of Array After K Negations

Problem: Given array and K, maximize sum after flipping sign of element K times.

Input: [2, -3, -1, 5], K=2

Output: 11

Explanation: Flip -3 → 3 and -1 → 1, sum = 2+3+1+5=11.

69. Binary Search on Rotated

Problem: Find target index in rotated sorted array (concept).

Input: [6,7,0,1,2,4,5], target=2

Output: 4

Explanation: Use modified binary search checking sorted half.

70. Count Set Bits in Range

Problem: Count total set bits in binary representations from 1 to N.

Input: N=3

Output: 4

Explanation: 1->1, 2->10, 3->11 total ones = 1+1+2=4.

71. Rotate Matrix Layers

Problem: Rotate matrix layers by k steps (conceptual).

Input: $[[1,2,3],[4,5,6],[7,8,9]]$, k=1

Output: $[[4,1,2],[7,5,3],[8,9,6]]$

Explanation: Extract each layer, rotate, place back.

72. Maximum Product Subarray

Problem: Find contiguous subarray with maximum product.

Input: [2,3,-2,4]

Output: 6

Explanation: Track max and min products due to negatives.

73. Shortest Palindrome by Adding Characters

Problem: Prepend minimum characters to make string palindrome (concept).

Input: "aacecaaa"

Output: "aacecaaa"

Explanation: Find longest palindromic prefix and mirror rest.

74. Evaluate Expression with Parentheses

Problem: Evaluate arithmetic expression with +, -, parentheses (no *, /).

Input: "(1+(4+5+2)-3)+(6+8)"

Output: 23

Explanation: Use stack to manage signs and partial results.

75. Count Balanced Brackets Substrings

Problem: Count balanced parentheses substrings length or number of valid substrings.

Input: "()()

Output: 2 (or total length 4 depending on spec)

Explanation: Scan with stack or counter to find valid segments.

76. Find Intersection of Two Arrays

Problem: Return intersection elements (unique) of two arrays.

Input: [1,2,2,3], [2,2]

Output: [2]

Explanation: Use sets to compute common elements.

77. Binary Gap

Problem: Longest distance between consecutive 1s in binary representation.

Input: 22 (10110)

Output: 2

Explanation: 10110 has gaps of lengths 2 and 1 → max 2.

78. Check If Two Strings Are Isomorphic

Problem: Determine if characters in two strings map one-to-one.

Input: "egg", "add"

Output: Yes

Explanation: Mapping e→a, g→d is consistent and bijective.

79. Count Inversions (conceptual)

Problem: Count number of inversion pairs in array ($i < j$ and $a[i] > a[j]$).

Input: [2,4,1,3,5]

Output: 3

Explanation: Use merge-sort based counting for $O(n \log n)$.

80. Check Subsequence

Problem: Determine if string s is subsequence of t.

Input: s="abc", t="ahbgdc"

Output: Yes

Explanation: Two-pointer scan through t to match s.

81. Find Majority Element > n/3

Problem: Find elements appearing more than $\lfloor n/3 \rfloor$ times.

Input: [1,1,1,3,3,2,2,2]

Output: [1,2]

Explanation: Extended Boyer-Moore generalization for $k=3$.

Problem 87: String Balance Check

Problem: Given a string containing various brackets, check if they are balanced. **Input:**

"{}[]{}" **Output:** Balanced **Explanation:** Every opening bracket has a corresponding closing bracket in correct order.

Problem 88: Missing Number Finder

Problem: An array of size N contains numbers from 1 to N+1 with one missing. Find the missing number. **Input:** [1, 2, 4, 5, 6] **Output:** 3 **Explanation:** Sum of first N natural numbers is 21, actual sum is 18, so missing number is 3.

Problem 89: Count Distinct Elements in Window

Problem: Given an array and window size K, count distinct elements in every window. **Input:** [1, 2, 1, 3, 4, 2, 3], K=4 **Output:** [3, 4, 4, 3] **Explanation:** Sliding windows show distinct element counts.

Problem 90: Majority Element

Problem: Find element appearing more than n/2 times. **Input:** [3, 3, 4, 2, 3, 3, 3] **Output:** 3 **Explanation:** 3 occurs 5 times, more than half of array size.

Problem 91: Rotate Array by K

Problem: Rotate array elements to the right by K positions. **Input:** [1, 2, 3, 4, 5, 6, 7], K=3 **Output:** [5, 6, 7, 1, 2, 3, 4] **Explanation:** Right rotation moves last K elements to the front.

Problem 92: Find First Repeating Element

Problem: Identify the first repeating element in an array. **Input:** [10, 5, 3, 4, 3, 5, 6] **Output:** 5 **Explanation:** 5 repeats before 3 when traversed from left.

Problem 93: Longest Common Prefix

Problem: Find the longest prefix common among all strings. **Input:** ["flower", "flow", "flight"] **Output:** "fl" **Explanation:** "fl" is common prefix to all.

Problem 94: Count Subarrays with Given Sum

Problem: Given array and integer K, count subarrays whose sum equals K. **Input:** [1, 1, 1], K=2 **Output:** 2 **Explanation:** Two subarrays [1,1] have sum 2.

Problem 95: Find Intersection of Two Arrays

Problem: Return unique intersection of two arrays. **Input:** [1,2,2,1], [2,2] **Output:** [2]
Explanation: 2 is common in both.

Problem 96: Move Zeroes to End

Problem: Move all 0s to end while keeping non-zero order same. **Input:** [0,1,0,3,12] **Output:** [1,3,12,0,0] **Explanation:** Shift non-zero elements left.

Problem 97: Check for Palindrome Number

Problem: Determine if number reads same forward and backward. **Input:** 121 **Output:** True
Explanation: 121 reversed is 121.

Problem 98: Merge Two Sorted Arrays

Problem: Merge two sorted arrays without duplicates. **Input:** [1,2,4], [1,3,4] **Output:** [1,2,3,4] **Explanation:** Combine and remove duplicates.

Problem 99: String Isomorphism

Problem: Check if two strings are isomorphic. **Input:** "egg", "add" **Output:** True
Explanation: Mapping e→a, g→d works consistently.

Problem 100: Find Pair with Given Sum

Problem: Check if array contains a pair with given sum K. **Input:** [10,15,3,7], K=17 **Output:** True
Explanation: $10 + 7 = 17$.

Problem 101: Longest Consecutive Sequence

Problem: Find longest sequence of consecutive integers. **Input:** [100,4,200,1,3,2] **Output:** 4
Explanation: Sequence [1,2,3,4] has length 4.

Problem 102: Product of Array Except Self

Problem: Return array where each element is product of all others. **Input:** [1,2,3,4] **Output:** [24,12,8,6] **Explanation:** Multiply all except self.

Problem 103: Subarray with Maximum Product

Problem: Find subarray having maximum product. **Input:** [2,3,-2,4] **Output:** 6 **Explanation:** Subarray [2,3] gives product 6.

Problem 104: Find All Anagrams

Problem: Find starting indices of anagrams of string p in s. **Input:** s="cbaebabacd", p="abc" **Output:** [0,6] **Explanation:** "cba" and "bac" are anagrams.

Problem 105: Maximum Subarray Sum (Kadane's Algorithm)

Problem: Find contiguous subarray with maximum sum. **Input:** [-2,1,-3,4,-1,2,1,-5,4] **Output:** 6 **Explanation:** [4,-1,2,1] gives max sum 6.

Problem 106: Check Rotation of String

Problem: Determine if one string is rotation of another. **Input:** s1="ABCD", s2="CDAB" **Output:** True **Explanation:** CDAB is rotation of ABCD.

Problem 107: Smallest Window Substring

Problem: Find minimum window in s containing all characters of t. **Input:** s="ADOBECODEBANC", t="ABC" **Output:** "BANC" **Explanation:** Smallest substring covering all chars.

Problem 108: Trapping Rain Water

Problem: Calculate water trapped between bars. **Input:** [0,1,0,2,1,0,1,3,2,1,2,1] **Output:** 6
Explanation: Water accumulates between taller bars.

Problem 109: Longest Palindromic Substring

Problem: Find longest substring that's palindrome. **Input:** "babad" **Output:** "bab" or "aba"
Explanation: Both are valid palindromes.

Problem 110: Container With Most Water

Problem: Find two lines forming container with max water. **Input:** [1,8,6,2,5,4,8,3,7]
Output: 49 **Explanation:** Lines at index 1 and 8 form largest area.

Problem 111: Count Inversions in Array

Problem: Count pairs (i, j) where $i < j$ and $\text{arr}[i] > \text{arr}[j]$. **Input:** [2, 4, 1, 3, 5] **Output:** 3
Explanation: Inversions are (2,1), (4,1), (4,3).

Problem 112: Minimum Platforms Required

Problem: Given train arrival and departure times, find min platforms needed. **Input:**
Arrival=[900,940,950,1100,1500,1800], Departure=[910,1200,1120,1130,1900,2000]
Output: 3 **Explanation:** At most 3 trains overlap at once.