Accolite-based Advance Coding Questions 2026 batch By – Mr. Durgesh StudyHub

Here's the Advanced Coding Test – Coding Round 2 (DSA + Hard Level) focused on DSA + Problem Solving

1. Subarray Sum Equals K

Problem:
Given an integer array and an integer `K`, find the total number of continuous subarrays whose sum equals `K`.

Input:

```
nums = [1,2,3]
K = 3
```

Output:

```
2
```

Explanation:
Subarrays `[1,2]` and `[3]` both sum to `3`.
Use prefix sum + hashmap.

---

2. Longest Substring Without Repeating Characters

Problem:
Find the length of the longest substring without repeating characters.

Input:

```
s = "abcabcbb"
```

Output:

```
3
```

Explanation:
Longest substring is `"abc"`.
Sliding window with hash set or map.

---

3. Product of Array Except Self

Problem:
Return an array where each element is the product of all elements except itself.

Input:

```
nums = [1,2,3,4]
```

Output:

```
[24,12,8,6]
```

Explanation:
No division allowed.
Use prefix and suffix products.

---

4. Rotate Matrix by 90 Degrees

Problem:
Rotate a square matrix by 90 degrees clockwise.

Input:

```
matrix =
1 2 3
4 5 6
```

```
7 8 9
```

Output:

```
7 4 1
8 5 2
9 6 3
```

Explanation:
First transpose, then reverse each row.

---

5. Find the First Missing Positive

Problem:
Find the smallest missing positive integer from an unsorted array.

Input:

```
nums = [3,4,-1,1]
```

Output:

```
2
```

Explanation:
Ignore negatives.
Place numbers at correct index (index-based hashing).

---

6. Majority Element (More than n/2 times)

Problem:
Find the element that appears more than `n/2` times.

Input:

```
nums = [2,2,1,1,1,2,2]
```

Output:

```
2
```

Explanation:
Use **Boyer–Moore Voting Algorithm**.

---

7. Validate Parentheses

Problem:
Check if the given string of brackets is valid.

Input:

```
s = "{[()]}"
```

Output:

```
true
```

Explanation:
Use stack to match opening and closing brackets.

---

8. Kth Largest Element in an Array

Problem:
Find the Kth largest element in an unsorted array.

Input:

```
nums = [3,2,1,5,6,4]
k = 2
```

Output:

```
5
```

Explanation:
Use min-heap of size `k` or Quick Select.

---

9. Merge Overlapping Intervals

Problem:
Merge all overlapping intervals.

Input:

```
intervals = [[1,3],[2,6],[8,10]]
```

Output:

```
[[1,6],[8,10]]
```

Explanation:
Sort intervals by start time, then merge greedily.

---

10. Longest Increasing Subsequence (LIS)

Problem:
Find the length of the longest increasing subsequence.

Input:

```
nums = [10,9,2,5,3,7,101,18]
```

Output:

```
4
```

Explanation:
Subsequence: `[2,3,7,101]`.
DP or Binary Search (O(n log n)).

---

11. Two Sum (Optimized)

Problem:
Given an array and a target, return indices of two numbers such that they add up to target.

Input:

```
nums = [2,7,11,15]
target = 9
```

Output:

```
[0,1]
```

Explanation:
Use hashmap to store visited numbers and check complement.

---

12. Longest Palindromic Substring

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

Input:

```
s = "babad"
```

Output:

```
"bab"
```

Explanation:
Expand around center for each character.

---

13. Container With Most Water

Problem:
Given heights of vertical lines, find max water container.

Input:

```
height = [1,8,6,2,5,4,8,3,7]
```

Output:

```
49
```

Explanation:
Two-pointer approach moving smaller height pointer.

---

14. Spiral Matrix Traversal

Problem:
Return elements of matrix in spiral order.

Input:

```
matrix =
1 2 3
4 5 6
7 8 9
```

Output:

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

Explanation:
Use boundaries (top, bottom, left, right).

---

15. Search in Rotated Sorted Array

Problem:
Search target in rotated sorted array.

Input:

```
nums = [4,5,6,7,0,1,2]
target = 0
```

Output:

```
4
```

Explanation:
Modified binary search.

---

16. Find Peak Element

Problem:
Find an element greater than its neighbors.

Input:

```
nums = [1,2,3,1]
```

Output:

```
2
```

Explanation:
Binary search approach; peak always exists.

---

17. Word Break (Boolean)

Problem:
Check if string can be segmented into dictionary words.

Input:

```
s = "leetcode"
dict = ["leet","code"]
```

Output:

```
true
```

Explanation:
DP to check valid segmentation.

---

18. Count Primes Less Than N

Problem:
Count prime numbers less than `n`.

Input:

```
n = 10
```

Output:

```
4
```

Explanation:
Use Sieve of Eratosthenes.

---

19. Minimum Size Subarray Sum

Problem:
Find minimal length subarray with sum ≥ target.

Input:

```
target = 7
nums = [2,3,1,2,4,3]
```

Output:

```
2
```

Explanation:
Sliding window technique.

---

20. Detect Cycle in Linked List

Problem:
Check whether linked list contains a cycle.

Input:

```
head = [3,2,0,-4], pos = 1
```

Output:

```
true
```

Explanation:
Use Floyd's Tortoise & Hare algorithm.

---

21. Find Duplicate Number

Problem:
Given an array of `n+1` integers where each integer is in range `[1, n]`, find the duplicate number.

Input:

```
nums = [1,3,4,2,2]
```

Output:

```
2
```

Explanation:
Use Floyd's Cycle Detection (slow & fast pointers).
No extra space allowed.

---

22. Set Matrix Zeroes

Problem:
If an element is `0`, set its entire row and column to `0`.

Input:

```
matrix =
1 1 1
1 0 1
1 1 1
```

Output:

```
1 0 1
0 0 0
1 0 1
```

Explanation:
Use first row & column as markers to get O(1) space.

---

23. Maximum Subarray Sum (Kadane's Algorithm)

Problem:
Find the contiguous subarray with the maximum sum.

Input:

```
nums = [-2,1,-3,4,-1,2,1,-5,4]
```

Output:

```
6
```

Explanation:
Subarray `[4,-1,2,1]` gives maximum sum.
Use Kadane's algorithm.

---

24. Check Anagram Strings

Problem:
Check whether two strings are anagrams of each other.

Input:

```
s = "listen"
t = "silent"
```

Output:

```
true
```

Explanation:
Count frequency of characters or sort both strings.

---

25. Binary Tree Level Order Traversal

Problem:
Return level-order traversal of a binary tree.

Input:

```
Tree: [3,9,20,null,null,15,7]
```

Output:

```
[[3],[9,20],[15,7]]
```

Explanation:
Use queue (BFS).

---

26. Minimum Difference Between Any Two Elements

Problem:
Find minimum absolute difference between any two elements.

Input:

```
nums = [3,8,15,1]
```

Output:

```
2
```

Explanation:
Sort array and check adjacent differences.

---

27. Reverse Words in a String

Problem:
Reverse the order of words in a string.

Input:

```
s = "the sky is blue"
```

Output:

```
"blue is sky the"
```

Explanation:
Split by spaces, reverse words, join again.

---

28. Subarray With Maximum XOR

Problem:
Find subarray having maximum XOR value.

Input:

```
nums = [8,1,2,12]
```

Output:

```
15
```

Explanation:
Use prefix XOR + Trie for optimization.

---

29. Balanced Binary Tree

Problem:
Check if a binary tree is height-balanced.

Input:

```
Tree = [3,9,20,null,null,15,7]
```

Output:

```
true
```

Explanation:
Height difference of left & right subtree ≤ 1 at every node.

---

30. Count Subarrays with Odd Sum

Problem:
Count number of subarrays with odd sum.

Input:

```
nums = [1,3,5]
```

Output:

```
4
```

Explanation:
Use prefix sum parity (even/odd count).

---

31. Longest Common Prefix

Problem:
Find the longest common prefix among an array of strings.

Input:

```
strs = ["flower","flow","flight"]
```

Output:

```
"fl"
```

Explanation:
Compare characters column-wise or sort strings and compare first & last.

32. Maximum Frequency Element

Problem:
Find the element that appears maximum times in an array.

Input:

```
nums = [1,3,2,1,4,1]
```

Output:

```
1
```

Explanation:
Use hashmap to store frequency counts.

---

33. Check Palindrome Linked List

Problem:
Check whether a singly linked list is a palindrome.

Input:

```
head = [1,2,2,1]
```

Output:

```
true
```

Explanation:
Find middle, reverse second half, then compare both halves.

---

34. Longest Consecutive Sequence

Problem:
Find length of the longest consecutive elements sequence.

Input:

```
nums = [100,4,200,1,3,2]
```

Output:

```
4
```

Explanation:
Sequence `[1,2,3,4]`.
Use HashSet for O(n) solution.

---

35. Count Vowels in All Substrings

Problem:
Count total vowels present in all substrings of a string.

Input:

```
s = "abc"
```

Output:

```
3
```

Explanation:
Each vowel contributes `(i+1)*(n-i)` times.

---

36. Check If Number Is Power of Two

Problem:
Check whether a given integer is power of 2.

Input:

```
n = 16
```

Output:

```
true
```

Explanation:
A power of 2 has only one set bit.

---

37. Next Greater Element

Problem:
For each element, find the next greater element to its right.

Input:

```
nums = [4,5,2,10]
```

Output:

```
[5,10,10,-1]
```

Explanation:
Use stack to maintain decreasing sequence.

---

38. Longest Subarray with Equal 0s and 1s

Problem:
Find the longest subarray containing equal number of 0s and 1s.

Input:

```
nums = [0,1,0,1,1,0]
```

Output:

```
6
```

Explanation:
Convert 0 → -1, then use prefix sum + hashmap.

---

39. Minimum Moves to Equal Array Elements

Problem:
Find minimum moves to make all array elements equal where one move increments n-1 elements by 1.

Input:

```
nums = [1,2,3]
```

Output:

```
3
```

Explanation:
Moves = sum(nums) − n × min(nums).

---

40. Binary Tree Maximum Depth

Problem:
Find maximum depth of a binary tree.

Input:

```
Tree = [3,9,20,null,null,15,7]
```

Output:

```
3
```

Explanation:
Use DFS recursion or BFS level counting.

---

1. Median of Two Sorted Arrays

Problem:
Given two sorted arrays, find the median of the combined dataset in **O(log(min(n,m)))** time.

Input:

```
nums1 = [1,3]
nums2 = [2]
```

Output:

```
2
```

Explanation:
Merged array → `[1,2,3]`.
Binary search on smaller array to partition correctly.

---

2. Trapping Rain Water

Problem:
Given elevation map, calculate total trapped rainwater.

Input:

```
height = [0,1,0,2,1,0,1,3,2,1,2,1]
```

Output:

```
6
```

Explanation:
Water trapped = min(leftMax, rightMax) − height[i].
Use two-pointer technique.

---

3. Longest Valid Parentheses

Problem:
Find length of longest valid parentheses substring.

Input:

```
s = ")()())"
```

Output:

```
4
```

Explanation:
Substring "()()" is valid.
Use stack or DP.

4. Minimum Window Substring

Problem:
Find smallest substring in `s` containing all characters of `t`.

Input:

```
s = "ADOBECODEBANC"
t = "ABC"
```

Output:

```
"BANC"
```

Explanation:
Sliding window + frequency map.

---

5. Largest Rectangle in Histogram

Problem:
Find largest rectangular area in histogram.

Input:

```
heights = [2,1,5,6,2,3]
```

Output:

```
10
```

Explanation:
Rectangle using bars `5, 6`.
Use stack to track previous smaller bars.

---

6. Word Ladder (Shortest Transformation)

Problem:
Find shortest transformation sequence length from `beginWord` to `endWord`.

Input:

```
begin = "hit"
end = "cog"
dict =
["hot","dot","dog","lot","log","co
g"]
```

Output:

```
5
```

Explanation:
BFS on word transformations (one letter change).

---

7. Maximum Sum Rectangle in 2D Matrix

Problem:
Find rectangle with maximum sum in a 2D matrix.

Input:

```
matrix =
1  2 -1
-3 4  5
```

Output:

```
9
```

Explanation:
Reduce 2D → 1D and apply Kadane's algorithm.

---

8. Detect Cycle in Directed Graph

Problem:
Check if a directed graph contains a cycle.

Input:

```
V = 4
Edges = [[0,1],[1,2],[2,3],[3,1]]
```

Output:

```
true
```

Explanation:
Use DFS + recursion stack or Kahn's Algorithm.

---

9. Kth Smallest Element in Sorted Matrix

Problem:
Find kth smallest element in sorted matrix.

Input:

```
matrix =
1  5  9
10 11 13
```

```
12 13 15
k = 8
```

Output:

```
13
```

Explanation:
Binary search on value range or min-heap.

---

10. Edit Distance

Problem:
Find minimum operations to convert one string into another.

Input:

```
word1 = "horse"
word2 = "ros"
```

Output:

```
3
```

Explanation:
Operations: delete, insert, replace.
Use DP table.

11. Sliding Window Maximum

Problem:
Given an array and window size `k`, find the maximum element in every sliding window.

Input:

```
nums = [1,3,-1,-3,5,3,6,7]
k = 3
```

Output:

```
[3,3,5,5,6,7]
```

Explanation:
Use deque to maintain decreasing order of indices.

---

12. Longest Increasing Path in a Matrix

Problem:
Find length of longest strictly increasing path in a matrix.

Input:

```
matrix =
9 9 4
6 6 8
2 1 1
```

Output:

```
4
```

Explanation:
Path: `1 → 2 → 6 → 9`.
Use DFS + memoization.

---

13. Count of Range Sum

Problem:
Count number of subarrays whose sum lies in `[lower, upper]`.

Input:

```
nums = [-2,5,-1]
lower = -2
upper = 2
```

Output:

```
3
```

Explanation:
Use prefix sums + modified merge sort.

---

14. Burst Balloons

Problem:
Maximize coins obtained by bursting balloons optimally.

Input:

```
nums = [3,1,5,8]
```

Output:

```
167
```

Explanation:
DP on intervals: last balloon burst gives max coins.

---

15. Shortest Path in Binary Matrix

Problem:
Find shortest path from top-left to bottom-right in binary matrix.

Input:

```
grid =
0 1
1 0
```

Output:

```
2
```

Explanation:
Use BFS (8-direction movement).

---

16. Minimum Number of Refueling Stops

Problem:
Find minimum refueling stops to reach target.

Input:

```
target = 100
```

```
startFuel = 10
stations =
[[10,60],[20,30],[30,30],[60,40]]
```

Output:

```
2
```

Explanation:
Use max-heap to refuel optimally.

---

17. Palindrome Partitioning (Minimum Cuts)

Problem:
Find minimum cuts so that every substring is palindrome.

Input:

```
s = "aab"
```

Output:

```
1
```

Explanation:
Partition as `"aa" | "b"`.
DP + palindrome check.

---

18. Alien Dictionary

Problem:
Given sorted dictionary of alien language, find character order.

Input:

```
["wrt","wrf","er","ett","rftt"]
```

Output:

```
"wertf"
```

Explanation:
Build graph and apply topological sorting.

---

19. Kth Largest Element in Stream

Problem:
Return kth largest element after each insertion.

Input:

```
k = 3
stream = [4,5,8,2]
```

Output:

```
[-1,-1,4,4]
```

Explanation:
Use min-heap of size `k`.

---

20. Maximum XOR of Two Numbers

Problem:
Find maximum XOR of any two numbers in array.

Input:

```
nums = [3,10,5,25,2,8]
```

Output:

```
28
```

Explanation:

---

1. Smallest Range Covering Elements from K Lists

Problem:
Given `k` sorted lists, find the smallest range that includes at least one number from each list.

Input:

```
[[4,10,15,24,26],
 [0,9,12,20],
 [5,18,22,30]]
```

Output:

```
[20,24]
```

Explanation:
Use min-heap + track current max.
Continuously shrink range while covering all lists.

---

2. Regular Expression Matching

Problem:
Implement regex matching with `.` and `*`.

Input:

```
s = "aab"
p = "c*a*b"
```

Output:

```
true
```

Explanation:
DP where `*` can represent zero or more of previous character.

---

3. Maximum Profit with K Transactions

Problem:
Find max profit with at most `k` stock transactions.

Input:

```
prices = [3,2,6,5,0,3]
k = 2
```

Output:

```
7
```

Explanation:
DP with states `(day, transactions,`

holding)
Optimized with rolling arrays.

---

4. Word Ladder II (All Shortest Paths)

Problem:
Return all shortest transformation sequences.

Input:

```
begin = "hit"
end = "cog"
dict =
["hot","dot","dog","lot","log","co
g"]
```

Output:

```
[
 ["hit","hot","dot","dog","cog"],
 ["hit","hot","lot","log","cog"]
]
```

Explanation:
BFS for shortest distance + DFS backtracking.

---

5. Longest Duplicate Substring

Problem:
Find the longest substring that appears at least twice.

Input:

```
s = "banana"
```

Output:

```
"ana"
```

Explanation:
Binary search on length + rolling hash (Rabin-Karp).

---

6. Trapping Rain Water II (2D)

Problem:
Given a 2D elevation map, compute trapped rainwater.

Input:

```
heightMap =
1 4 3
```

```
3 2 5
```

Output:

```
2
```

Explanation:
Min-heap BFS from boundary inward.

---

7. Minimum Cost to Cut a Stick

Problem:
Given stick length and cut positions, find minimum total cost.

Input:

```
n = 7
cuts = [1,3,4,5]
```

Output:

```
16
```

Explanation:
DP on intervals — choose optimal first cut.

---

8. Shortest Superstring

Problem:
Find the shortest string containing all given strings.

Input:

```
["alex","loves","leetcode"]
```

Output:

```
"alexlovesleetcode"
```

Explanation:
DP with bitmask + overlap computation.

---

9. Maximum Number of Non-Overlapping Subarrays with Sum = Target

Problem:
Find maximum number of non-overlapping subarrays with given sum.

Input:

```
nums = [1,1,1,1,1]
```

```
target = 2
```

Output:

```
2
```

Explanation:
Greedy with prefix sum reset when target reached.

---

10. Number of Ways to Form a Target String Given a Dictionary

Problem:
Count ways to form target using characters column-wise from words.

Input:

```
words = ["acca","bbbb","caca"]
target = "aba"
```

Output:

```
6
```

Explanation:
DP + frequency precomputation per column.

11. Minimum Number of Taps to Open to Water a Garden

Problem:
Given `n` and an array `ranges`, find the minimum number of taps to water the entire garden `[0, n]`.

Input:

```
n = 5
ranges = [3,4,1,1,0,0]
```

Output:

```
1
```

Explanation:
Convert to interval coverage and apply greedy jump-game style expansion.

12. Maximum Sum of Rectangle No Larger Than K

Problem:
Find the maximum sum of any rectangle in a 2D matrix such that it is ≤ `k`.

Input:

```
matrix = [[1,0,1],[0,-2,3]]
k = 2
```

Output:

```
2
```

Explanation:
Compress columns → 1D array → prefix sums with ordered set (BST).

---

13. Count Different Palindromic Subsequences

Problem:
Count distinct palindromic subsequences in a string.

Input:

```
s = "bccb"
```

Output:

```
6
```

Explanation:
DP with tracking previous/next occurrence of characters to avoid duplicates.

---

14. Skyline Problem

Problem:
Given buildings, return the skyline formed by them.

Input:

```
buildings =
[[2,9,10],[3,7,15],[5,12,12]]
```

Output:

```
[[2,10],[3,15],[7,12],[12,0]]
```

Explanation:
Sweep line + max-heap to track active heights.

15. Find Critical and Pseudo-Critical Edges in MST

Problem:
Find edges that are critical or pseudo-critical in the MST.

Input:

```
n = 5
edges = [...]
```

Output:

```
critical = [0,1]
pseudo = [2,3]
```

Explanation:
Compute MST with and without each edge using Kruskal.

16. Longest Increasing Subsequence in 2D (Russian Doll Envelopes II)

Problem:
Max number of envelopes that can be nested.

Input:

```
[[5,4],[6,4],[6,7],[2,3]]
```

Output:

```
3
```

Explanation:
Sort by width ↑ and height ↓, then LIS on heights.

---

17. Shortest Path with Alternating Colors

Problem:
Find shortest path where edges must alternate colors.

Input:

```
n = 3
```

```
redEdges = [[0,1]]
blueEdges = [[1,2]]
```

Output:

```
[0,1,2]
```

Explanation:
BFS with state `(node, lastColor)`.

---

18. Make Array Strictly Increasing

Problem:
Replace elements with values from another array to make it strictly increasing.

Input:

```
arr1 = [1,5,3,6,7]
arr2 = [1,3,2,4]
```

Output:

```
1
```

Explanation:
DP + binary search on sorted `arr2`.

---

19. Maximize Grid Happiness

Problem:
Place introverts/extroverts on grid to maximize happiness.

Input:

```
m = 2, n = 3
introverts = 1
extroverts = 2
```

Output:

```
240
```

Explanation:
State compression DP with row masks.

---

20. Minimum Cost to Connect Two Groups of Points

Problem:
Connect two groups minimizing total cost.

Input:

```
cost = [[15,96],[36,2]]
```

Output:

```
17
```

Explanation:
Bitmask DP ensuring all points in both groups are connected.

---

21. Cherry Pickup (Grid DP)

Problem:
Two people start at `(0,0)` and move to `(n-1,n-1)` simultaneously, collecting cherries. Find the maximum cherries collected.

Input:

```
grid =
0 1 -1
1 0 -1
1 1  1
```

Output:

```
5
```

Explanation:
Both move together → convert to **3D DP** or `(r1,c1,r2)` state.
Avoid double counting cherries.

---

22. Minimum Number of Days to Eat N Oranges

Problem:
You can eat 1 orange, or if divisible by 2 eat `n/2`, or by 3 eat `2n/3`.
Find minimum days to eat all oranges.

Input:

```
n = 10
```

Output:

```
4
```

Explanation:
Use **DP + memoization** with greedy division when possible.

---

23. Count of Smaller Numbers After Self

Problem:
For each element, count smaller numbers to its right.

Input:

```
nums = [5,2,6,1]
```

Output:

```
[2,1,1,0]
```

Explanation:
Modified **merge sort** or **Fenwick Tree**.

---

24. Maximum Score Words Formed by Letters

Problem:
Choose subset of words to maximize score using available letters.

Input:

```
words = ["dog","cat","dad","good"]
letters =
["a","a","c","d","d","d","g","o","o"]
```

Output:

```
23
```

Explanation:
Backtracking + frequency pruning (subset DP).

---

25. Remove Boxes

Problem:
Remove boxes to maximize score; score = `k*k` for removing `k` equal boxes together.

Input:

```
boxes = [1,3,2,2,2,3,4,3,1]
```

Output:

```
23
```

Explanation:
Very hard DP: `dp(l, r, k)` = max score when `k` same boxes are attached to left.

26. Split Array Largest Sum

Problem:
Split array into `k` subarrays to minimize the largest sum.

Input:

```
nums = [7,2,5,10,8]
k = 2
```

Output:

```
18
```

Explanation:
Binary search on answer + greedy validation.

---

27. Minimum Number of Increments on Subarrays to Form Target Array

Problem:
Build target array using increment operations on subarrays.

Input:

```
target = [1,2,3,2,1]
```

Output:

```
3
```

Explanation:
Total operations = sum of all positive differences.

---

28. Parallel Courses III

Problem:
Courses have prerequisites and durations. Find minimum time to complete all.

Input:

```
n = 3
relations = [[1,3],[2,3]]
time = [3,2,5]
```

Output:

```
8
```

Explanation:
DAG + **topological DP** (longest path).

29. Maximum AND Sum of Array

Problem:
Place numbers into `k` slots to maximize AND-sum.

Input:

```
nums = [1,2,3,4]
numSlots = 2
```

Output:

```
6
```

Explanation:
Bitmask DP over slots (each slot can take max 2 numbers).

---

30. Strange Printer

Problem:
A printer prints same characters in one turn.
Find minimum turns to print the string.

Input:

```
s = "aba"
```

Output:

```
2
```

Explanation:
Interval DP: merge same characters to reduce turns.
