🚀 100 Logic-Building Questions (Newbie → Pro)

🟢 Level 1: Basics & Foundations (1–20)

Focus: Loops, conditionals, arrays, strings, simple math.

1. Print the first n natural numbers.
2. Reverse a number.
3. Check if a number is palindrome.
4. Find factorial of n.
5. Fibonacci sequence up to n.
6. Sum of digits of a number.
7. Check prime number.
8. Print all primes up to n.
9. GCD & LCM of two numbers.
10. Count vowels and consonants in a string.
11. Reverse a string.
12. Check if a string is palindrome.
13. Frequency of characters in a string.
14. Find largest element in an array.
15. Find second largest element in an array.
16. Find smallest element in an array.

17. Check if array is sorted.
18. Linear search in an array.
19. Binary search in a sorted array.
20. Rotate array by k steps.

🟡 Level 2: Intermediate Problems (21–40)

Focus: Sorting, hashing, two pointers, sliding window.

21. Implement bubble sort.
22. Implement selection sort.
23. Implement insertion sort.
24. Merge two sorted arrays.
25. Implement merge sort.
26. Implement quick sort.
27. Count duplicates in an array.
28. Find intersection of two arrays.
29. Find union of two arrays.
30. Longest subarray with sum k.
31. Find majority element (> n/2).
32. Two-sum problem.
33. Move all zeros to the end.
34. Kadane's algorithm (max subarray sum).
35. Trapping rain water.
36. Stock buy and sell (max profit).
37. Check anagram strings.
38. Find longest palindrome substring.
39. Count substrings with equal 0s and 1s.
40. Sliding window maximum.

🔵 Level 3: Data Structures (41–60)

Focus: Stacks, queues, linked list, recursion.

41. Implement stack using array.
42. Implement queue using array.
43. Implement circular queue.

44. Implement stack using 2 queues.
45. Implement queue using 2 stacks.
46. Reverse a stack using recursion.
47. Next greater element.
48. Balanced parentheses check.
49. Evaluate postfix expression.
50. Implement singly linked list.
51. Reverse a linked list.
52. Detect cycle in linked list.
53. Find middle of linked list.
54. Merge two sorted linked lists.
55. Remove duplicates from linked list.
56. Implement doubly linked list.
57. Nth node from end in linked list.
58. Implement min stack.
59. LRU cache implementation.
60. Tower of Hanoi problem.

🔴 Level 4: Advanced DS & Trees (61–80)

Focus: Trees, heaps, graphs, dynamic programming.

61. Binary tree traversals (inorder, preorder, postorder).
62. Level order traversal.
63. Height of a binary tree.
64. Diameter of a binary tree.
65. Lowest common ancestor (LCA).
66. Serialize and deserialize a binary tree.
67. Check if two trees are identical.
68. Mirror of a binary tree.
69. Convert sorted array to BST.
70. BST search and insertion.
71. Delete a node in BST.
72. Heapify (min-heap/max-heap).
73. Heap sort.
74. Kth largest element using heap.
75. Median of a running stream.
76. Implement graph using adjacency list.
77. BFS traversal.
78. DFS traversal.
79. Detect cycle in graph.
80. Dijkstra’s algorithm.

🟣 Level 5: Competitive Programming (81–100)

Focus: Greedy, DP, backtracking, bit manipulation, advanced logic.

81. Subset sum problem.
82. Coin change problem.
83. 0/1 Knapsack problem.
84. Longest common subsequence.
85. Longest increasing subsequence.
86. Matrix chain multiplication.
87. Word break problem.
88. N-Queens problem.
89. Sudoku solver.
90. Rat in a maze problem.
91. Count islands in a grid.
92. Minimum spanning tree (Kruskal/Prim).
93. Topological sort.
94. Bellman-Ford algorithm.
95. Floyd-Warshall algorithm.
96. Maximum bipartite matching.
97. Trie implementation (insert, search).
98. Maximum XOR pair using Trie.
99. Segment tree (range sum query).
100. Fenwick tree (Binary Indexed Tree).

✅ By solving these in order, you'll move from **basic programming** → **problem-solving** → **DSA mastery** → **CP-level logic**.