DATA STRUCTURES & ALGORITHMS

ARRAYS & STRINGS

Array Basics

1. What is an array? What is a linear data structure?
2. What is the time complexity for access, insert, and delete in an array?
3. What is the difference between static array and dynamic array?
4. What are the methods to search for an element in an array?
5. What is the two pointer technique? Give an example.
6. What is the sliding window technique?
7. How do you reverse an array? Show both in-place and extra space methods.
8. How do you find the largest and smallest element in an array?
9. How do you find duplicate elements in an array?
10. How do you merge two arrays (sorted)?

Array Problems (Easy)

11. Find the missing number in an array (one missing from 1 to n).
12. Rotate an array (left and right rotation).
13. Find leaders in an array (all smaller elements on the right side).
14. Move all zeros to the end of the array.
15. Find a pair with a given sum (Two Sum problem).
16. Find maximum subarray sum (Kadane's Algorithm).
17. Find the majority element in an array (appears > n/2 times).
18. Best time to buy and sell stock.
19. Rearrange array alternately (max, min, second max, second min).
20. Find subarray with given sum.

String Problems

21. How do you reverse a string?
22. Check if a string is a palindrome.
23. Check if two strings are anagrams.
24. Find the first non-repeating character.
25. Find the longest substring without repeating characters.

SEARCHING & SORTING

Searching Algorithms

26. What is linear search? What is its time complexity?
27. What is binary search? When can you use it?
28. What is the time complexity of binary search and why?
29. Implement recursive binary search.
30. Find the first and last occurrence in a sorted array.
31. Search in a rotated sorted array.
32. Find a peak element in an array.
33. Search in a 2D matrix (sorted rows and columns).

Sorting Algorithms

34. Explain bubble sort algorithm with time complexity.
35. What is selection sort?
36. What is the use case of insertion sort?
37. Explain merge sort algorithm (Divide and Conquer).
38. What is quick sort? What is a pivot element?
39. What is the difference between merge sort and quick sort?
40. Which sorting algorithm is stable and why?

LINKED LISTS

Linked List Basics

41. What is a linked list? How is it different from an array?
42. What is the difference between singly linked list and doubly linked list?
43. What is a circular linked list?
44. What are the methods to insert a node in a linked list (beginning, end, middle)?
45. How do you delete a node from a linked list?
46. What is the time complexity for operations on a linked list?

Linked List Problems

47. How do you reverse a linked list?
48. Detect a cycle in a linked list (Floyd's algorithm).
49. Find the middle element in a linked list.

50. Merge two sorted linked lists.
51. Remove the nth node from the end of a linked list.
52. Check if a linked list is a palindrome.
53. Find the intersection point of two linked lists.
54. Remove duplicates from a sorted linked list.
55. Add two numbers represented by linked lists.

STACKS & QUEUES

Stack

56. What is a stack? Explain the LIFO principle.
57. What are the basic operations of a stack (push, pop, peek)?
58. How do you implement a stack using an array?
59. Implement a stack using a linked list.
60. Check for valid parentheses using a stack.
61. Find the next greater element using a stack.
62. Convert infix to postfix.
63. Evaluate a postfix expression.

Queue

64. What is a queue? What is the FIFO principle?
65. What are the basic operations of a queue (enqueue, dequeue, front, rear)?
66. What is a circular queue?
67. Implement a queue using two stacks.
68. Implement a stack using two queues.

TREES

Tree Basics

69. What is a tree? Define root, parent, child, and leaf nodes.
70. What is a binary tree?
71. What is a Binary Search Tree (BST)? What are its properties?
72. What are the types of tree traversal?
73. What is inorder traversal? (Left, Root, Right)

74. Explain preorder traversal. (Root, Left, Right)
75. What is postorder traversal? (Left, Right, Root)
76. How do you implement level order traversal (BFS)?

Tree Problems

77. Find the maximum depth/height of a binary tree.
78. Check if two trees are identical.
79. Check if a tree is balanced (height-balanced).
80. Find the diameter of a binary tree.
81. Find the Lowest Common Ancestor (LCA) in a BST.
82. Check if a tree is a BST or not.
83. Find the inorder successor in a BST.
84. Convert a sorted array to a BST.
85. Serialize and deserialize a binary tree.

HASHING & HASH TABLES

Hashing Concepts

86. What is a hash table? What is a hash function?
87. What is a collision? What are collision resolution techniques?
88. What is the difference between chaining and open addressing?
89. What is the time complexity for operations on a hash table?
90. How is a Dictionary/HashMap implemented internally?

Hashing Problems

91. Solve the two sum problem using a hash map.
92. Group anagrams together using hashing.

ADVANCED TOPICS

Recursion & Dynamic Programming

93. What is recursion? Why is the base case important?
94. Fibonacci series - both recursive and iterative approaches.
95. Calculate factorial recursively.
96. Explain the Tower of Hanoi problem.

97. What is Dynamic Programming? Memoization vs Tabulation.
98. What is the 0/1 Knapsack problem?

Graph Basics

99. What is a graph? What is the difference between directed and undirected graphs?
100. What are BFS (Breadth First Search) and DFS (Depth First Search)?