Sure, here’s a comprehensive list of 100 coding questions commonly asked in technical interviews. They cover a range of topics including algorithms, data structures, and problem-solving techniques.

### Array and String Problems

1. \*\*Two Sum\*\*: Find two numbers in an array that add up to a target value.

2. \*\*Reverse an Array\*\*: Reverse the elements of an array.

3. \*\*Rotate an Array\*\*: Rotate an array to the right by `k` steps.

4. \*\*Merge Sorted Arrays\*\*: Merge two sorted arrays into one sorted array.

5. \*\*Find the Duplicate Number\*\*: Given an array of integers where each integer is between `1` and `n`, find the duplicate number.

6. \*\*Find the Missing Number\*\*: Find the missing number in a sequence from `1` to `n`.

7. \*\*Move Zeroes\*\*: Move all zeroes in an array to the end while maintaining the order of non-zero elements.

8. \*\*Product of Array Except Self\*\*: Given an array of integers, return an array where each element is the product of all other elements.

9. \*\*Find the Longest Substring Without Repeating Characters\*\*: Determine the length of the longest substring without repeating characters.

10. \*\*Check for Anagram\*\*: Determine if two strings are anagrams of each other.

### Linked List Problems

11. \*\*Reverse a Linked List\*\*: Reverse a singly linked list.

12. \*\*Detect a Cycle in a Linked List\*\*: Determine if a linked list has a cycle.

13. \*\*Find the Middle of a Linked List\*\*: Find the middle node of a linked list.

14. \*\*Merge Two Sorted Linked Lists\*\*: Merge two sorted linked lists into one sorted linked list.

15. \*\*Remove N-th Node From End of List\*\*: Remove the n-th node from the end of a linked list.

16. \*\*Add Two Numbers Represented by Linked Lists\*\*: Add two numbers represented by linked lists.

### Stack and Queue Problems

17. \*\*Valid Parentheses\*\*: Determine if a string containing parentheses is valid.

18. \*\*Implement a Stack Using Queues\*\*: Implement a stack using two queues.

19. \*\*Implement a Queue Using Stacks\*\*: Implement a queue using two stacks.

20. \*\*Evaluate Reverse Polish Notation\*\*: Evaluate an expression given in Reverse Polish Notation.

### Tree Problems

21. \*\*Binary Tree Inorder Traversal\*\*: Perform inorder traversal of a binary tree.

22. \*\*Binary Tree Preorder Traversal\*\*: Perform preorder traversal of a binary tree.

23. \*\*Binary Tree Postorder Traversal\*\*: Perform postorder traversal of a binary tree.

24. \*\*Check if a Tree is Balanced\*\*: Determine if a binary tree is balanced.

25. \*\*Find the Height of a Binary Tree\*\*: Find the height of a binary tree.

26. \*\*Lowest Common Ancestor of a Binary Tree\*\*: Find the lowest common ancestor of two nodes in a binary tree.

27. \*\*Convert a Binary Tree to a Doubly Linked List\*\*: Convert a binary tree to a doubly linked list.

28. \*\*Check if a Binary Tree is a Subtree of Another\*\*: Check if a binary tree is a subtree of another binary tree.

29. \*\*Binary Search Tree (BST) Validation\*\*: Check if a binary tree is a valid binary search tree.

30. \*\*Find the K-th Smallest Element in a BST\*\*: Find the k-th smallest element in a binary search tree.

### Graph Problems

31. \*\*Depth-First Search (DFS)\*\*: Implement depth-first search on a graph.

32. \*\*Breadth-First Search (BFS)\*\*: Implement breadth-first search on a graph.

33. \*\*Detect a Cycle in a Graph\*\*: Determine if a graph contains a cycle.

34. \*\*Find Shortest Path in a Graph\*\*: Find the shortest path between two nodes in a graph (Dijkstra’s algorithm).

35. \*\*Topological Sort\*\*: Perform topological sorting of a directed graph.

36. \*\*Find Strongly Connected Components\*\*: Find strongly connected components in a directed graph (Kosaraju’s algorithm).

37. \*\*Minimum Spanning Tree\*\*: Find the minimum spanning tree of a graph (Kruskal’s or Prim’s algorithm).

38. \*\*Count Number of Islands\*\*: Count the number of islands in a 2D grid of water and land.

39. \*\*Word Ladder\*\*: Find the shortest transformation sequence from one word to another using a dictionary.

40. \*\*Clone a Graph\*\*: Clone an undirected graph.

### Dynamic Programming Problems

41. \*\*Knapsack Problem\*\*: Solve the 0/1 knapsack problem using dynamic programming.

42. \*\*Longest Common Subsequence\*\*: Find the longest common subsequence between two strings.

43. \*\*Longest Increasing Subsequence\*\*: Find the length of the longest increasing subsequence in an array.

44. \*\*Edit Distance\*\*: Find the minimum number of operations required to convert one string to another.

45. \*\*Coin Change Problem\*\*: Find the minimum number of coins required to make a given amount.

46. \*\*Maximum Subarray Sum\*\*: Find the contiguous subarray with the maximum sum.

47. \*\*Matrix Chain Multiplication\*\*: Find the minimum number of multiplications needed to multiply a chain of matrices.

48. \*\*Partition Equal Subset Sum\*\*: Determine if an array can be partitioned into two subsets with equal sum.

49. \*\*Unique Paths in a Grid\*\*: Find the number of unique paths from the top-left corner to the bottom-right corner of a grid.

50. \*\*0/1 Knapsack Problem\*\*: Solve the knapsack problem with binary choices.

### Sorting and Searching Problems

51. \*\*Binary Search\*\*: Implement binary search on a sorted array.

52. \*\*Quick Sort\*\*: Implement quicksort to sort an array.

53. \*\*Merge Sort\*\*: Implement merge sort to sort an array.

54. \*\*Find Peak Element\*\*: Find a peak element in an array where an element is greater than its neighbors.

55. \*\*Find the First and Last Position of an Element in a Sorted Array\*\*: Find the first and last position of a given element in a sorted array.

56. \*\*Find the K-th Smallest Element in an Array\*\*: Find the k-th smallest element in an unsorted array.

57. \*\*Sort Colors\*\*: Sort an array containing `0s`, `1s`, and `2s` (Dutch National Flag problem).

58. \*\*Find Missing Range\*\*: Find missing ranges in a sorted array of integers.

59. \*\*Find Median of Two Sorted Arrays\*\*: Find the median of two sorted arrays.

60. \*\*Find the Majority Element\*\*: Find the element that appears more than `n/2` times in an array.

### Bit Manipulation Problems

61. \*\*Single Number\*\*: Find the single number that appears only once in an array where every other number appears twice.

62. \*\*Counting Bits\*\*: Count the number of `1` bits in the binary representation of an integer.

63. \*\*Power of Two\*\*: Determine if a number is a power of two.

64. \*\*Reverse Bits\*\*: Reverse the bits of a given 32-bit unsigned integer.

65. \*\*Bitwise AND of Numbers Range\*\*: Find the bitwise AND of all numbers in a given range.

### Math Problems

66. \*\*Factorial of a Number\*\*: Compute the factorial of a non-negative integer.

67. \*\*Prime Number Check\*\*: Determine if a number is a prime number.

68. \*\*Greatest Common Divisor (GCD)\*\*: Find the greatest common divisor of two numbers (Euclidean algorithm).

69. \*\*Sieve of Eratosthenes\*\*: Implement the Sieve of Eratosthenes to find all prime numbers up to a given limit.

70. \*\*Fibonacci Sequence\*\*: Compute the nth Fibonacci number.

### Backtracking Problems

71. \*\*N-Queens\*\*: Solve the N-Queens problem.

72. \*\*Sudoku Solver\*\*: Solve a Sudoku puzzle.

73. \*\*Permutations\*\*: Find all permutations of a given list of numbers.

74. \*\*Combination Sum\*\*: Find all combinations of numbers that sum up to a target value.

75. \*\*Subset Sum Problem\*\*: Find subsets of an array that sum up to a given value.

### Greedy Problems

76. \*\*Activity Selection\*\*: Select the maximum number of activities that can be performed by a single person.

77. \*\*Fractional Knapsack Problem\*\*: Solve the fractional knapsack problem where you can take fractions of items.

78. \*\*Job Sequencing Problem\*\*: Find the maximum profit schedule of jobs.

79. \*\*Minimum Number of Coins\*\*: Find the minimum number of coins required to make a given amount.

80. \*\*Huffman Coding\*\*: Implement Huffman coding for data compression.

### Miscellaneous Problems

81. \*\*LRU Cache\*\*: Implement an LRU (Least Recently Used) cache.

82. \*\*Design a Stack With Min Function\*\*: Implement a stack that supports retrieving the minimum element.

83. \*\*Design a Data Structure That Supports Insert, Delete, GetRandom in Constant Time\*\*: Implement a data structure that supports O(1) operations.

84. \*\*Count and Say\*\*: Generate the `n`th term of the count-and-say sequence.

85. \*\*Spiral Order Matrix\*\*: Print elements of a matrix in spiral order.

86. \*\*Find the Peak Element\*\*: Find a peak element in an array.

87. \*\*Find All Anagrams in a String\*\*: Find all starting indices of anagrams of a given string in another string.

88. \*\*Valid Sudoku\*\*: Determine if a given Sudoku board is valid.

89. \*\*Add Binary\*\*: Add two binary numbers represented as strings.

90. \*\*Implement Trie (Prefix Tree)\*\*: Implement a trie with insert, search, and startsWith operations.

91. \*\*Median of a Stream\*\*: Find the median of a stream of

integers.

92. \*\*Find the Missing Ranges\*\*: Find missing ranges in a sorted array of integers.

93. \*\*Minimum Window Substring\*\*: Find the smallest window in a string that contains all characters of another string.

94. \*\*Largest Rectangle in Histogram\*\*: Find the largest rectangle area in a histogram.

95. \*\*Count Islands\*\*: Count the number of islands in a grid.

96. \*\*Reverse Words in a String\*\*: Reverse the words in a given string.

97. \*\*Add Two Numbers\*\*: Add two numbers represented by linked lists.

98. \*\*Find the Longest Palindromic Substring\*\*: Find the longest palindromic substring in a string.

99. \*\*Rotate Image\*\*: Rotate an image represented by an `n x n` matrix by 90 degrees.

100. \*\*Unique Paths\*\*: Find the number of unique paths in a grid from the top-left corner to the bottom-right corner.

Practicing these questions will help you prepare for a variety of technical interviews and improve your problem-solving skills.