Step 1: Core Foundations (Priority 1)
These are the most common topics you'll encounter in coding interviews. Master these first.

1. Arrays and Strings

- Start with basic array manipulations.
- Practice:
  - Traversals, insertions, deletions.
  - Sorting (Bubble, Selection, Merge, Quick Sort).
  - Searching (Linear and Binary Search).
  - String operations (reversal, anagram checks, substring finding).
- **Problems to Solve:**
  - Two Sum
  - Longest Substring Without Repeating Characters
  - Rotate Array
  - Maximum Subarray (Kadane's Algorithm)

2. Hashing

- Learn how to use HashMaps and HashSets.
- Practice:
  - Frequency counting.
  - Checking duplicates.
  - Solving problems like two-sum efficiently.
- **Problems to Solve:**
  - Subarray Sum Equals K
  - Longest Consecutive Sequence
  - Group Anagrams

3. Stacks and Queues

- Understand their properties and when to use them.
- Practice:
  - Next Greater Element
  - Valid Parentheses
  - Implement Stack Using Queues
- **Problems to Solve:**
  - Min Stack
  - Sliding Window Maximum

Step 2: Intermediate Data Structures (Priority 2)
These topics build on the basics and add versatility.

4. Linked Lists

- Master operations like insertion, deletion, reversal, and cycle detection.
- Practice:
  - Merge Two Sorted Lists
  - Reverse a Linked List
  - Detect Cycle in a Linked List (Floyd's Algorithm)
  - Remove N-th Node from End of List

5. Trees

- Learn basic tree traversals (inorder, preorder, postorder).
- Understand Binary Search Tree (BST) properties.
- Practice:
    - Maximum Depth of Binary Tree
    - Lowest Common Ancestor
    - Serialize and Deserialize Binary Tree
- Gradually move to:
    - Diameter of Binary Tree
    - Balanced Binary Tree

6. Recursion and Backtracking

- Practice:
    - Permutations and Combinations
    - N-Queens Problem
    - Sudoku Solver
    - Subset Sum Problem

Step 3: Advanced Data Structures (Priority 3)

These are essential for solving complex problems.

7. Graphs

- Understand representation (Adjacency Matrix and List).
- Master:
    - BFS and DFS (recursive and iterative).
    - Shortest Path Algorithms (Dijkstra, Bellman-Ford).
    - Minimum Spanning Trees (Prim's and Kruskal's Algorithms).
- **Problems to Solve:**
    - Clone Graph
    - Number of Islands
    - Detect Cycles in Undirected/Directed Graphs

8. Tries

- Learn how to use Tries for prefix-based problems.
- **Problems to Solve:**
    - Implement Trie (Prefix Tree)
    - Word Search II
    - Auto-complete System

9. Dynamic Programming (DP)

- Start with simple problems:
    - Fibonacci Sequence
    - Climbing Stairs
    - Longest Increasing Subsequence
    - Knapsack Problem
- Progress to:
    - Longest Palindromic Substring
    - Edit Distance
    - Minimum Path Sum

Step 4: Combination and Optimization (Priority 4)
Start combining multiple concepts and optimizing solutions.

10. Sliding Window and Two Pointers
- Useful for array and string problems.
- **Problems to Solve:**
  - Longest Substring Without Repeating Characters
  - Minimum Window Substring
  - Trapping Rain Water

11. Greedy Algorithms
- Understand when greedy strategies work and when they fail.
- **Problems to Solve:**
  - Activity Selection Problem
  - Huffman Encoding
  - Minimum Number of Platforms

12. Divide and Conquer
- Master solving problems by breaking them into sub-problems.
- **Problems to Solve:**
  - Merge Sort
  - Quick Sort
  - Median of Two Sorted Arrays

Step 5: Master Problem-Solving
Once you've learned individual topics:
1. **Mix Problems:** Solve problems involving multiple data structures and algorithms.
   - Example: Use heaps and graphs together for Dijkstra's Algorithm.
2. **Simulate Interviews:** Solve problems under a time constraint to simulate real interviews.
3. **Analyze Your Code:** Optimize for time and space complexity.

Suggested Weekly Plan

Week	Focus Topic	Goals
1	Arrays, Strings, and Hashing	Solve 20–30 problems.
2	Linked Lists and Recursion	Solve 15–20 problems.
3	Stacks, Queues, and Trees	Solve 20 problems with tree traversals.
4	Graphs and BFS/DFS	Solve 15 graph problems.
5	Dynamic Programming and Tries	Solve 20–30 DP problems.
6	Mixed Problems and Optimization	Practice interview-level problems.

Would you like specific resources or problem recommendations for a particular topic?