Contents

Coding Interview Questions & Study Plan

Overview

This document contains a comprehensive 2-week study plan for preparing for technical coding interviews. The focus is on reinforcing high-yield LeetCode-style problem patterns that emphasize clean problem-solving, edge case handling, and clear communication rather than obscure algorithms.

Target Audience: Software engineers, engineering managers, and technical leads preparing for coding interviews at any company.

Study Plan Structure

- **Duration:** 2 weeks (14 days)

- **Daily Commitment:** 60–90 minutes
- **Difficulty Focus:** Easy to Medium level problems
- **Primary Goal:** Build fluency in solving problems with clear logic and clean code
- **Secondary Goal:** Develop strong communication skills for technical interviews

---

Week 1: Core Patterns & Fundamentals

Day 1: Arrays & Hash Tables

Focus: Basic data structure manipulation and lookup optimization

- ☒ Two Sum - Hash map for O(1) lookups
- ☒ Contains Duplicate - Set operations

Key Concepts: Hash tables, time/space complexity trade-offs

Day 2: String Manipulation

Focus: Character processing and sliding window techniques

- ☒ Valid Anagram - Character frequency counting
- ☒ Longest Substring Without Repeating Characters - Sliding window

Key Concepts: Sliding window, character maps, string traversal

Day 3: Intervals & Merging

Focus: Range processing and overlap detection

- ☒ Merge Intervals - Interval merging logic
- ☒ Insert Interval - Insertion with merging

Key Concepts: Sorting by intervals, overlap detection, merging logic

Day 4: Binary Trees

Focus: Tree traversal and basic tree operations

- ☒ Invert Binary Tree - Simple tree recursion
- ☒ Binary Tree Level Order Traversal - BFS traversal
- ☒ Diameter of Binary Tree - Track max depth during recursion

Key Concepts: Recursion, tree traversal (DFS/BFS), queue usage

Day 5: Recursion & Backtracking

Focus: Recursive problem solving and state exploration

- ☒ Generate Parentheses - Valid combinations
- ☒ Permutations - Generate all permutations recursively

Key Concepts: Backtracking, state space exploration, recursive thinking

Day 6: Practice & Mock Interview

Focus: Applying learned patterns under time pressure

- ☐ Select 1–2 problems from Days 1–5
- ☐ Time yourself (45 minutes max)
- ☐ Practice explaining your approach out loud
- ☐ Focus on edge cases and testing

Day 7: Review & Consolidation

Focus: Strengthening weak areas and pattern recognition

- ☐ Re-solve the most challenging problems from Days 1–5
- ☐ Practice whiteboard coding (pen and paper)
- ☐ Review time/space complexity analysis

Week 2: Advanced Patterns & Interview Readiness

Day 8: Sorting & Heaps

Focus: Priority-based algorithms and efficient sorting

- ☒ Kth Largest Element in an Array - Quickselect or heap
- ☒ Top K Frequent Elements - Heap operations

Key Concepts: Priority queues, heap operations, greedy algorithms

Day 9: Graph Algorithms

Focus: Graph traversal and connected components

- ☒ Number of Islands - DFS/BFS on grid
- ☒ Clone Graph - Graph copying with visited tracking
- ☒ Course Schedule - Topological sort using BFS

Key Concepts: DFS, BFS, visited tracking, graph representation

Day 10: Linked Lists

Focus: Pointer manipulation and list operations

- ☒ LRU Cache - Implement with dict and doubly linked list
- ☒ Reverse Linked List - Pointer reversal

Key Concepts: Two pointers, list reversal, edge case handling

Day 11: Mathematical & Edge Cases

Focus: Problem-solving with mathematical insights

- ☒ Missing Number - Mathematical approach
- ☒ Two Sum II – Input Array Is Sorted - Two pointers technique

Key Concepts: Mathematical optimization, modular arithmetic, edge case handling

Day 12: Mock Interview Session

Focus: Full interview simulation

- ☐ Conduct a timed 45-60 minute session
- ☐ Choose a new Medium-level problem
- ☐ Practice the complete interview flow: clarification → approach → coding → testing
- ☐ Record yourself or practice with a friend

Day 13: Speed & Fluency

Focus: Building confidence and speed

- ☐ Re-solve 3 previously completed problems from scratch
- ☐ Focus on implementation speed while maintaining code quality
- ☐ Practice explaining solutions concisely

Day 14: Final Preparation

Focus: Interview strategy and system design warmup

- ☐ Review your personal problem-solving framework
- ☐ Practice one system design problem (e.g., "Design a URL shortener")
- ☐ Prepare questions to ask your interviewer

Interview Best Practices

Problem-Solving Framework

1. **Clarify Requirements**
   - Ask clarifying questions about inputs, outputs, and constraints
   - Confirm edge cases and assumptions
   - Understand the expected time/space complexity
2. **Plan Your Approach**
   - Think out loud and explain your strategy
   - Start with a brute force solution, then optimize
   - Discuss trade-offs between different approaches
3. **Write Clean Code**
   - Use meaningful variable names
   - Structure your code logically with helper functions
   - Handle edge cases explicitly
4. **Test Your Solution**
   - Walk through your code with the given example
   - Test edge cases (empty input, single element, etc.)
   - Consider time and space complexity

Communication Tips

- **Verbalize your thought process** throughout the interview
- **Ask questions** when requirements are unclear
- **Explain trade-offs** when choosing between approaches
- **Stay calm** if you get stuck; think out loud and work through it step by step
- **Be honest** about areas where you're uncertain

Common Patterns to Master

- **Two Pointers:** For sorted arrays, palindromes, or finding pairs
- **Sliding Window:** For substring problems or array subarrays
- **Hash Maps:** For fast lookups and counting
- **DFS/BFS:** For tree and graph traversal
- **Dynamic Programming:** For optimization problems with overlapping subproblems
- **Binary Search:** For sorted data or optimization problems

---

Additional Resources

Practice Platforms

- LeetCode - Primary platform for coding practice
- HackerRank - Alternative practice platform
- CodeSignal - Interview-style challenges

Study Materials

- Cracking the Coding Interview - Classic interview prep book
- Elements of Programming Interviews - Comprehensive problem collection
- AlgoExpert - Video explanations and curated problems

Mock Interview Platforms

- Pramp - Free peer-to-peer mock interviews
- Interviewing.io - Anonymous mock interviews with engineers
- CodeSignal Interview Practice - Automated coding assessments

---

Progress Tracking

Use this section to track your progress and notes:

Completed Problems

- ☐ Day 1: Arrays & Hash Tables
- ☐ Day 2: String Manipulation

- ☐ Day 3: Intervals & Merging
- ☐ Day 4: Binary Trees

☐ Day 5: Recursion & Backtracking
☐ Day 6: Practice & Mock Interview
☐ Day 7: Review & Consolidation
☐ Day 8: Sorting & Heaps
☐ Day 9: Graph Algorithms
☐ Day 10: Linked Lists
☐ Day 11: Mathematical & Edge Cases
☐ Day 12: Mock Interview Session
☐ Day 13: Speed & Fluency
☐ Day 14: Final Preparation

Personal Notes

Use this space to note patterns you struggled with, insights you gained, or areas that need more practice.

Good luck with your coding interviews!