Beginner’s Guide to Preparing for Technical Interviews

Subtitle: Master coding problems, CS fundamentals, and strategies to confidently crack developer interviews.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app

Introduction

Technical interviews test your **problem-solving skills, coding ability, and understanding of computer science concepts**. This guide provides beginners with a structured approach to prepare efficiently and improve their chances of success.

Step 1: Understand the Interview Format

- **Coding Round:** Solve algorithmic problems in a coding language
- **System Design / Architecture (for advanced roles)**
- **Behavioral Questions:** Teamwork, projects, challenges
- **Online Assessments:** Platforms like HackerRank, LeetCode, CodeSignal

---

Step 2: Strengthen Core CS Concepts

- **Data Structures:** Arrays, Strings, Linked Lists, Stacks, Queues, Trees, Graphs, Hash Maps
- **Algorithms:** Sorting, Searching, Recursion, Dynamic Programming, Greedy algorithms
- **Time & Space Complexity:** Understand Big O notation
- **Practice:** Implement each data structure and algorithm from scratch

---

Step 3: Learn a Primary Programming Language

- Pick **one language** (e.g., JavaScript, Python, Java)
- Focus on:
    - Syntax

- Functions / OOP concepts
- Standard libraries for arrays, strings, etc.

Step 4: Practice Coding Problems

- Start with **easy problems**, then gradually move to medium and hard
- Platforms: LeetCode, HackerRank, Codewars
- Daily target: Solve **1–2 problems consistently**
- Track problems in categories: Arrays, Strings, Trees, Graphs, DP

Step 5: Mock Interviews

- Use platforms like Pramp or Interviewing.io
- Practice explaining your thought process aloud
- Record yourself to improve communication and clarity

Step 6: Common Problem Types & Examples

- **Arrays / Strings:** Reverse array, find duplicates, palindrome check
- **Linked Lists:** Reverse a linked list, detect cycles
- **Trees / Graphs:** DFS, BFS, find max depth
- **Dynamic Programming:** Fibonacci, knapsack, coin change
- **Practice Tip:** Understand patterns instead of memorizing solutions

---

Step 7: System Design Basics (Optional for Beginners)

- Learn about: Scalability, caching, databases
- Start small: Design URL shortener, basic chat app
- Focus on explaining your approach clearly

Step 8: Behavioral Questions Preparation

- STAR Method: Situation, Task, Action, Result
- Common topics:

.

- ◦ Projects you built
- ◦ Challenges faced and solutions
- ◦ Teamwork and leadership

Step 9: Daily Preparation Routine

1. Solve 1–2 coding problems
2. Review previous mistakes
3. Read about 1 CS concept
4. Mock interview or explain solution aloud
5. Repeat consistently

Step 10: Key Takeaways

- Consistency is more important than intensity
- Understand core CS concepts and data structures
- Practice coding problems daily
- Mock interviews improve confidence and communication
- Document your progress and review mistakes regularly

Visit **haas.dev** for technical interview guides, coding challenges, and structured preparation resources for beginner developers.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app
