THE UNIVERSITY OF CHICAGO | CAREER ADVANCEMENT

Technical Interview Prep Guide

Preparation Strategies

How to Approach Interview Prep

Preparing for interviews is one of the most important and overlooked parts of the tech recruiting process. Whether you're gearing up for technical interviews that require months of focused practice or behavioral interviews that ask you to tell compelling stories about your experience, preparation is what sets strong candidates apart.

In a competitive job market, you rarely “wing it” into a tech role. Practicing early helps build confidence, sharpen your problem-solving skills, and reduce anxiety when real interviews arrive.

It’s important to approach this process with the mindset of a learner, not a perfectionist. Interviews are about showing how you think, communicate, and approach challenges, and are not about whether you have all the answers. With the right strategy and preparation, interviews become less of a test and more of a conversation where you get to showcase what you bring to the table.

Interviews aren't just about proving you're the right fit for a company; they're also about discovering if the company is the right fit for you. Instead of seeing interviews as exams, think of them as two-way conversations where you're not just showcasing your strengths but also evaluating how the company's culture, values, and opportunities align with your own goals.

When to Start

For most students or early-career candidates aiming for software engineering roles, especially at larger tech companies, 3 to 6 months of consistent, focused preparation is a realistic and effective timeline for studying algorithms and technical interview paradigms.

Many aspects of behavioral interview preparation can be done in advance (as you'll see with the Step 4 assignment), but fine-tuning to tailor your talking points to meet the expectations of the hiring team for a particular company or role can be done once you've been notified of an interview by a recruiter or hiring manager.

Resources:

https://interviewkickstart.com/blogs/articles/how-long-does-it-take-to-prepare-for-coding-interview

https://www.techinterviewhandbook.org/coding-interview-study-plan/

Technical Interviews

Types of Technical Interviews

Online Assessments (OAs)

A timed, automated coding test often used as the first filter in the recruiting process. It usually includes 1–3 algorithmic problems on platforms like HackerRank or Codility, completed without real-time feedback.

Hiring Manager Interviews

A 30–60 minute conversation with a hiring manager or engineer to evaluate your technical fit and past experience. May include light technical questions, project walkthroughs, or assessing your approach to real-world tasks.

Example Questions:

- Can you explain the time and space complexity of a binary search algorithm?
- Imagine a service you've built is returning errors. Walk me through your troubleshooting process.
- Would you choose a relational or non-relational database for [example use case]? Why?
- How have you applied a data structure or algorithm you learned in class to a project or real problem?

Coding Interviews

Live interview where you solve algorithmic or data structure problems in real time, typically on a whiteboard or collaborative editor. Interviewers evaluate your problem-solving process, coding ability, and communication.

Clarify the question

- Do you understand what the question is asking?
- Is there additional information that'd be helpful?
    - E.g. "Can the numbers be repeated?"

Design a solution

- Don't start coding immediately
- Start with the first solution that comes to mind, then iterate
- Describe your algorithms and performances
- Make sure you handle edge cases
- Think out loud
    - You should be communicating with the interviewer during all of these steps
    - Refine and improve solution collaboratively with your interviewer
    - It's OK to talk about ideas you later realize don't work

Write your code

- Break your solution into smaller parts using helper functions
- If you don't know the exact API, ask your interviewer
- Write the code in the language and style you're most comfortable with
- Feel free to clarify if they'd like you to write full code or pseudocode

Test your code

- Don't assume your code works
- Walkthrough your code line by line, using your test cases from earlier. Run through at least 2 of the test cases to check your work.

- If you realize there are ways to optimize your solution even more, talk about it!

System Design Interviews

The objective of system design interviews is to evaluate a candidate's skill at designing real-world software systems involving multiple components. At the entry-level, these focus more on foundational thinking than on building complex architectures. Interviewers want to see how you break down problems, identify components, and communicate your reasoning.

Example Questions:

- Design a basic chat application
    - Tests: message delivery, persistence, real-time vs. async trade-offs, user sessions
- Design a leaderboard for a game
    - *Tests:* sorting/ranking logic, read/write performance, updating scores in real time
- Design a search autocomplete feature
    - *Tests:* prefix trees (tries), caching, handling user input at scale

Take Home Assignments

Instead of (or in addition to) live interviews, some companies give you a practical project to complete on your own time so that you can present your work and approach in subsequent interviews.

Example prompt:

Imagine you're the Associate Product Manager for a mobile calendar app used by students and working professionals. Your team is considering adding a “smart scheduling” feature that suggests meeting times based on calendar availability, time zones, and user preferences.

Write a short product brief that outlines your thinking about problem definition, goals and success metrics, feature scope, tradeoff and prioritization, and collaboration and stakeholders.

How to Prepare

1. *Learn the fundamentals*
    a. Make sure you understand data structures (arrays, hash maps, trees, graphs) and algorithms (sorting, recursion, dynamic programming, etc.).
2. *Practice regularly*
    a. Use platforms like LeetCode, HackerRank, or CodeSignal to solve 2–3 problems per week at first, then ramp up as interviews approach.
3. *Focus on patterns, not just solutions*
    a. Study common problem types and build mental templates (e.g. sliding window, binary search, BFS/DFS). Recognizing patterns saves time.
4. *Say it out loud*
    a. Explain your thought process while solving problems. This mirrors real interviews and helps you communicate clearly under pressure.

Resources:

Evaluation criteria

- https://www.techinterviewhandbook.org/coding-interview-rubrics/
- https://brianjenney.medium.com/the-ai-penalty-is-real-coding-interviews-just-got-weirder-97d1606a1a90

Preparation tools

- https://www.techinterviewhandbook.org/coding-interview-cheatsheet/
- https://github.com/anushka23g/Complete-Placement-Preparation#readme
- https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-interview-questions
- https://www.teamblind.com/blog/the-blind-75-guide-the-ultimate-coding-interview-prep-resource/
- https://leetcode.com/problem-list/xoqag3yj/

Company-specific materials

- Amazon
- Meta
- Microsoft
- NVIDIA
- Google

Behavioral Interviews

Don’t Skip Behavioral Prep

Technical skills might get you the interview, but strong communication, compelling stories, and a clear understanding of your motivations are what help you land the offer.

Behavioral interviews are your chance to show how you think, work with others, and solve problems. Employers use them to predict how you’ll perform based on your past experiences. They're also assessing whether you’re a good fit for the team relative to the role’s expectations. Most companies identify specific “signal areas," or key attributes or competencies they believe matter for success, and use structured interviews to reduce bias and make the process more equitable.

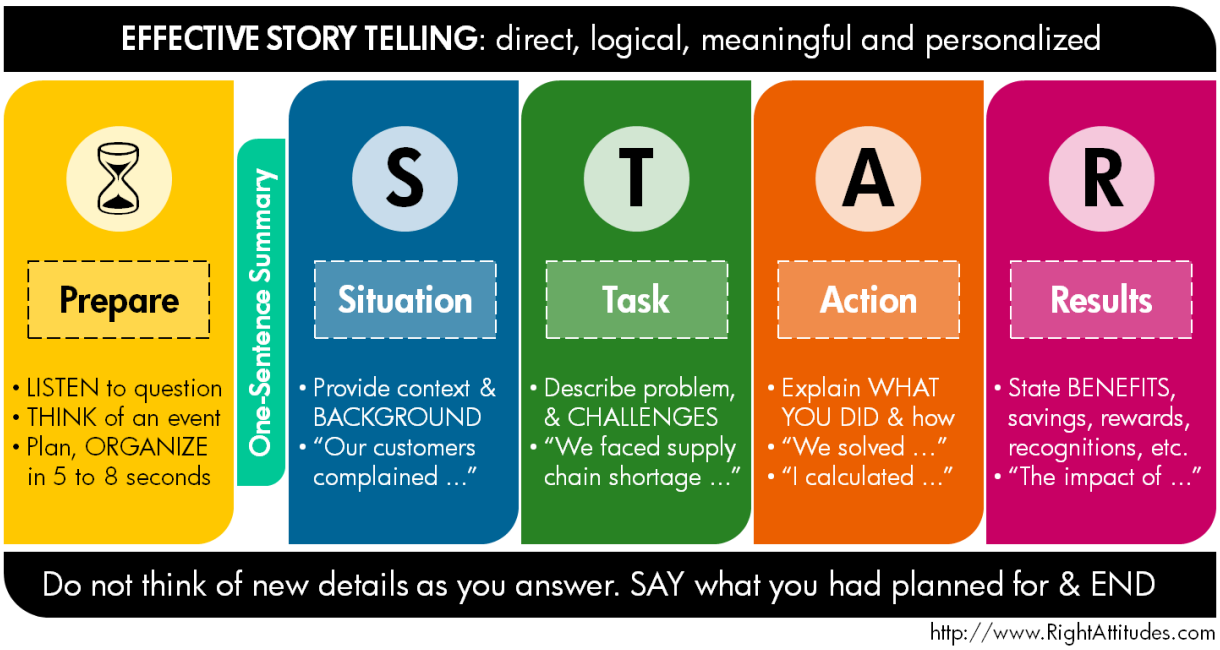
Nearly every tech company includes at least one behavioral round for software engineers. These interviews usually focus on:

- Talking through specific experiences on your resume
- Providing examples that show core attributes (e.g. teamwork, leadership, problem-solving)
- Sharing your career goals and motivations

Use the STAR format to structure your answers

The **STAR** method helps you answer questions about past experiences in a clear and compelling way:

- **S**ituation: Set the scene. What was happening, and why were you involved?
- **T**ask: What goal were you working toward? Include scope, stakes, or specific outcomes.
- **A**ction: What did you do? Walk through your decisions and thought process.
- **R**esult: What happened, and what did you learn? Include metrics if possible.



What if I don't have traditional work experience?

What matters is how you demonstrate your skills.

- Talk about class projects that involved problem-solving or collaboration
- Share what you learned from internships, even if they were short-term
- Highlight leadership or initiative in student orgs, volunteer work, or campus jobs
- Point to skills you've built on your own through online courses, certifications, or independent learning

If you're asked something you don't have direct experience with, try one of these responses:

- "I can share an example from a project I worked on that's relevant."
- "Here's how I would approach it based on what I've learned."
- "In my student org leadership role, I've handled similar situations."
- "I developed this skill through an online course, and here's how I'd apply it."

Common mistakes to avoid

Rambling

- Keep your answers concise, ideally under two minutes.
- Use 1–2 sentences to set up the story, then focus on what you did.

- If the interviewer wants more detail, they'll ask.

Underselling yourself

- This is your moment to showcase your impact.
- Don't downplay your accomplishments, own them.
- Especially in team projects, use "I" to describe your specific contributions.

Being inauthentic

- Avoid cliché answers like "my biggest weakness is that I work too hard."
- Focus on real stories that show self-awareness and growth.
- No one expects perfection, they want to know if you're someone they'd want to work with.

Not answering the question

- If you're not prepared, it's easy to drift off-topic.
- Practice ahead of time so you're ready to respond clearly, even in areas where you feel less confident.

Resources:

https://docs.google.com/document/d/1bqxSajTaElsJaHUzMTNG8ny7-SM9GrU-tGLpEOx-UkU/edit?usp=sharing

https://thebehavioral.substack.com/p/roadmap-to-behavioral-interview-prep