# Common Issues In Coding Interviews

And how to fix them

## The biggest, scariest issues

### I keep getting lost or stuck in the middle of technical questions.

Getting stuck during a coding interview can be really demoralizing. That is, until you get good at getting un-stuck. That's right, you can get *good* at *getting un-stuck*! You just have to learn the steps (/tricks-for-getting-unstuck-programming-interview).

But surprisingly, sometimes you're *supposed* to get stuck, and sometimes you're *supposed* to lose your train of thought. To understand why, read up on how the coding interview is like a maze (/why-youre-hitting-dead-ends-in-whiteboard-interviews)

Of course, with more practice you're less likely to get stuck or lose your train of thought. Check out our practice coding interview questions (/all-questions).

Finally, make sure you're doing everything you can to get yourself into the best possible headspace in the 24 hours before your big interview (/24-hours-before-onsite-whiteboard-coding-interview).

### It takes me forever to solve a single problem.

The trick to finishing problems faster is using a specific *process* and sticking to it:

1. **Brainstorm and design your algorithm by manipulating sample inputs by hand on the whiteboard**. Don't start writing code until you know exactly how your algorithm will work.

2. **Code it up as quickly as possible**. Don't get caught up in details like, "should this be a '<' or a '<='?"—just make a checkmark in the margin and move on. Don't start debugging it until it's all written out.
3. **Finally, walk through your code with a sample input and fix any bugs you find.**

The important lesson here is to never skip ahead. Only move on to the next step after finishing the last step. This keeps your thinking more organized, makes it easier for your interviewer to follow what you're doing, helps you avoid mistakes, and ultimately makes you move faster.

This process is explained in more detail in our general coding interview tips article (/coding-interview-tips).

## I'm practicing, but I'm not getting better.

You're probably making a common mistake with how your practice is structured. Is this you?:

You pull up a practice problem, work on it, get as far as you can . . . until you get stuck. You give up and look at the answer.

"*Whaaat?? How do you even come up with something like that?*"

You understand *what the answer is*, but you have no idea *how you could have gotten there yourself*.

*That's* the common mistake. Simply looking at questions and answers isn't enough. You need to learn *how to derive those answers yourself*.

That's why with Interview Cake, each of our questions (/all-questions) walks you through the thought process for coming up with the answer, including a review at the end where we pull out the generalizable lessons that can be applied to solving *other* questions.

Whether or not you use our full course, there are plenty of other small things you can do to make sure you're getting the most out of your coding interview practice (/getting-the-most-from-coding-interview-practice-sessions).

## I don't have a CS degree. I don't understand big O notation and algorithms.

A lot of people struggle with data structures, algorithms, and big O notation (/article/big-o-notation-time-and-space-complexity). Especially people who don't have a computer science degree.

It's easy to think this stuff is just objectively *hard to understand*, since it's associated with the "academic" side of software. That makes it seem more technical and difficult.

**The truth is this stuff just *feels* technical and difficult because people are bad at teaching it.**

Yes, thinking in algorithms and data structures is a specific skill that's different from general coding. It's a separate thing you have to learn.

But it's very learnable. Check out our Intuitive Guide to Data Structures and Algorithms (/data-structures-and-algorithms-guide).

### I . . . feel like I'm just not good at this stuff :/

This feeling is very common. The interview process makes us doubt ourselves. It eats away at our confidence. This is called impostor syndrome, and it can be fixed (/impostor-syndrome-in-programming-interviews).

## The rest of the job search process

### How do I *get* interviews?

### But I don't know the latest hip new framework or language.

### How long should I allow for my job search?

### I got an offer but it expires soon! What do I do?

### What about behavioral questions? How do I prepare for those?

# Practicing

**I know I should practice, but I have trouble finding the time.**

***How* should I practice?**

**How long should I spend on each each practice problem?**

**Should I do mock interviews? How?**

# Onsite Interviews

**What should I do the day before an onsite interview?**

**My whiteboard always gets really messy :/**

# Miscellaneous

**What do I do if I get rejected?**

Want more coding interview help?

Check out **interviewcake.com** for more advice, guides, and practice questions.