Gmail **rishav ghosh <rishavghosh605@gmail.com>**

A trick for getting good at coding interviews FASTER

1 message

Parker from Interview Cake <yourfriends@interviewcake.com> Wed, May 15, 2019 at 2:49 PM
To: rishavghosh605@gmail.com

This is Day 1/7 of our 1-week coding interview email course.

I figured this out while coaching my dear friend Alice through her job search. That experience, by the way, inspired me to start Interview Cake!

Alice had fallen on hard times and was crashing on my couch. She needed a new job, and wanted to shoot for her first *real* software engineering position. But she had worries that she wasn't qualified. She wasn't a computer science major in college, and she was weak on data structures and algorithms.

Long story short, with some focused practice we got her *really good* at coding interviews. She landed a few offers, and ended up choosing Facebook. She went from crashing on my couch to making more money than I was.

In running through practice problems with Alice, I had a realization: the hard part of coding interviews is having that breakthrough "Aha!" moment.

The secret to those "Aha" moments? They *always* come from applying a simple *algorithmic pattern*.

What's an "algorithmic pattern"? They take on a few different forms. But there are only so many of them. Some examples:

- Go bottom-up instead of top-down.
- Cut the problem in half and solve each half.
- Use a stack.

Of course, pattern recognition happens naturally, in our subconscious. That's how learning works. But with deliberate effort, it happens *way faster*.

That's what Alice and I did. We focused our attention on learning the *patterns*. Suddenly we weren't just learning answers—we were learning *approaches*. And we started covering a *lot* more ground.

So here's the tip: start collecting your own list of "patterns." Start a new doc on your computer or—even better—grab a fresh notebook. For every practice question you do, take a few minutes at the end to remember the moments where you got stuck. The things you had trouble figuring out. The things you got wrong at first. Basically, what *approaches* you learned, that you can *apply to future questions*.

You should have just a few of these per question—at least 1, and no more than 5.

After each practice session, review your *whole* list of patterns. Read it at the *beginning* of each practice session too. Heck, read it once real quick before bed. This'll add a nice layer of rigor to your practice, so you're really internalizing the lessons you're learning.

Why aren't I just *giving* you the patterns? It wouldn't be as effective. The patterns don't *really* make sense until you've put them in your own words by discovering them in the context of a real coding interview question. Trust me.

That said, tomorrow I'll share a few of the patterns *I've* found to be most helpful :)

I know we’re just getting started, but hopefully you’re finding these tips useful so far. As long as you follow along, I think you’ll have a pretty solid foundation for interviews by the end of the week.

Our full coding interview prep course is a great option if you’re looking to capitalize on that momentum and really start building up your pattern list, too. Remember: it only costs money if it works (if you don't get the job, I'll give you your money back).

Any questions? Just wanna vent about some coding interview stress? Get in touch by replying to this email.

Later,
Parker