Practicing Code Interview Questions is like Studying for the Exam

Written by Seth Corker on Benevolent Bytes

Getting into tech is popular at the moment. It seems as though it's easier than ever to start learning and get on track with breaking into the tech sector. Whether you want to be an engineer, quality assurance, product manager, CEO or anything in-between, there are countless resources available online. Books and bootcamps, podcasts and personalised courses; you name it, it probably exists. Having a wealth of quality knowledge is amazing. It's great to see a variety of different approaches to teaching on so many interesting topics but I've become aware of more courses specializing on how to pass a coding interview. Learning how to pass a coding interview reminds me of when, in high school I studied at the last minute to pass a science exam. I passed but I couldn't tell you anything about the course beyond the first few months. It didn't help me gain an understanding of the subject, it gave me enough surface level knowledge to answer questions in the exam. That's it. Services offering to teach you how to ace the interview are missing the point but are only part of the problem. Companies are responsible for setting expectations. Testing applicants in this arbitrary manner is not a good method to identify skill.

The problem with coding interviews

A programmer learns a broad range of topics when graduating from a traditional computer science or software engineering programme. The same topics that seem to come up in coding interviews. So, as an interviewer, if an applicant can answer questions related to O(n) notation or navigating a binary search tree comfortably, they have some idea of what they are talking about. It's probable they have the skills to back it up. This isn't always the case but it should be a good indication. If you're lacking experience and looking to get a job as a developer but you're studying to pass the interview, it seems like an attempt to cheat the system. The danger is that you get past the first hurdle only to fall flat on your face when you encounter the actual job. I don't agree with common interview questions but, tricking an interviewer into believing you're more capable than actuality is dangerous.

The idea of a technical interview where you have to recall obscure computer science trivia and jargon might not be a useful measure. It focuses on specialised knowledge that favours particular types of people. People who are capable of holding facts and algorithms in their head rise to the top. It's not a great measure of skill and when used incorrectly it can be entirely meaningless. Don't expect an applicant to have deep knowledge how to implement binary search or use Dijkstra's algorithm if you're hiring for a frontend engineer. If you're going to interview someone, make it relevant to the role. It's good for a backend engineer to have an idea of how frontend works but they shouldn't be tested on the intricacies of positioning elements in CSS. The thought process of trying to solve a real problem might be far more valuable.

An approach as an interviewer

Take a look at recent tickets in your issue tracker that you or your team has worked on recently. Find one that could be a useful problem to solve in an interview context.

1. Pick a ticket that is typical of the role you're hiring for
2. Make sure it's somewhat challenging or can be approached in different ways Give some background about the project and context needed to solve the issue and ask the candidate how they might go about solving the issue.

- How did they approach the problem?
- Did they identify a need for more context?
- What techniques do they use?
- Are they able to identify possible pitfalls in their approach?
- Tell them how the issue was solved in practice. Do they think there could be improvements or pitfalls of the approach taken?

Interviews that value the interviewee and focus on problem solving with relevance to the day-to-day job are far more valuable for everyone involved. Make your questions relevant to the position you want to fill to really find a good fit for the role. If a candidate doesn't know how to solve a given problem, why? Do they have the skill-set but are under pressure? Do they have strengths in other areas?

How to actually ace the interview

It's important to learn about the position you want to fulfill and identify your strengths and weaknesses with regard to the role. If you have some gaps in your knowledge then work on filling them. Ask yourself:

- How important do you this knowledge is for the role?
- Can you brush up on it or does it require some more thorough learning?

If it's a small gap then you might be able to pass the interview regardless. If it's a deep well that you're missing, it's best to focus on learning or identifying this gap in the interview. Don't study just for the interview! Make sure you have a good understanding of concepts and brush up if you need to but fumbling your way through the interview is not the answer. You're not learning the subject, you're learning how to pass the test.

I hope that this has been useful in some way and if you're deciding to take the leap into the world of tech, don't put all your focus on the interview. Focus on learning the fundamentals so if you get pas the interview you know you're in the right place. It helps if you have a body of work to show too. Remember that the interview is a hurdle but it circumventing it doesn't help anyone.