C++ Daily Interview

Nathan Warner

Northern Illinois
University

Computer Science
Northern Illinois University
United States

Contents

The different interview processes

1.1 The big tech style interview process

One of them, the more usual one nowadays is very competitive and has several rounds. It starts with an initial screening call that is often conducted by a non-technical person, a recruiter. However, I've heard and seen cases where even the first contacts were made by engineers so that they can see earlier whether you'd be a good fit for the team.

This first round might have been preceded by a 0th round with some takeaway exercise. The idea behind is that if you cannot prove a certain level of expertise, they don't want to waste their time on the applicant. Sometimes this exercise is way too long and many of us would outright reject such homework. If it's more reasonable, you can complete it in an hour or two.

After the screening, there is a technical round that is usually quite broad and there is not a lot of time to go deep on the different topics. It's almost sure that you'll get an easy or medium-level Leetcode-style coding exercise. For those, you must be able to reason about algorithmic complexities and it's also very useful if you are more than familiar with the standard library of your language. Apart from a coding exercise, expect more general questions about your chosen language.

By understanding your language deeper - something this book helps with - you'll have a better chance to reach the next and usually final round of interviews, the so-called on-site. Even if it's online, it might still be called on-site and it's a series of interviews you have to complete in one day or sometimes spanned over one day.

It typically has 3 or 4 different types of interviews.

- Behavioural interviews focusing on your soft skills
- A system design interview where you get a quite vague task to design a system. You have to clarify what the requirements are and you have to come up with the high-level architecture and dig deeper into certain parts
- There are different kinds of coding interviews

    1. **Coding exercises**: You won't be able to solve coding exercises with what you learn in this book, but you'll be able to avoid some pitfalls with a deeper understanding of the language. Daily C++ Interview helps you to achieve that understanding. In addition, you must practice on pages like Leetcode, Hackerrank, Codingame, etc
    2.
    3. **Debug interview**: You receive a piece of code and you have to find the bugs. Sometimes this can be called a code review interview. It's still about finding bugs. Personally, I find it a bit deeper than a simple coding exercise. In a coding interview, you are supposed to talk about design flaws, code smells, and testability. If you know C++ well enough, if you try to answer some of the questions of Daily C++ Interview on a day-to-day basis, you'll have a much better chance to recognize bugs, smells, flaws and pass the interview.

1.2 The shorter interview process

Certain companies try to compete for talent by shortening their interview cycles and making a decision as fast as possible. Often they promise a decision in less than 10 days. Usually, they don't offer so competitive packages - but even that is not always true - so they try to compete on something else.

A shorter decision cycle obviously means fewer interviews. Sometimes this approach is combined with the lack of coding interviews - at least for senior engineers. The idea behind is that many engineers despise the idea of implementing those coding exercises. They find it irrelevant and even derogative to implement a linked list. Instead, they will ask questions that help evaluate how deep you understand the given programming language.

Categories

The questions in this book fall into the following categories

- Auto and type deduction
- Keyword static and its different uses
- Polymorphism, inheritance, and virtual functions
- Lambda functions
- How to use const
- Best practices in modern c++
- Smart pointers
- References, universal references
- C++20
- Special function and the rules of how many
- OOP, inheritance, poylmorphism
- Observable behaviors
- The STL
- Misc