Ankush Khanna
M.Sc. Computer Science
Accolite Interview Experience - 2020

Online Round: MCQ Test

This round was conducted on EduThrill. There were 30 MCQ questions and 30 minutes in total. The problems were based on Computer Science Fundamentals (CSF), mostly from Data Structures, Algorithms, Operating Systems, DBMS, Networking. Also, we had basic math and aptitude questions. There wasn't any negative marking. The questions were fairly easy to do. Just basic knowledge would do.

Online Round: Coding Test

This round was conducted on Codelyzer. There was one problem and one hour in total.

Problem
Given a list of non negative integers, check whether it can be partitioned into 3 mutually exclusive and exhaustive sets with equal sum.

https://www.techiedelight.com/3-partition-problem/

Advice

1) Think more, compile less. Platform's latency was very high.
2) Stay calm and positive. If you can't figure out the solution, think about the closest problems you know, they might help.

Technical Round 1

It was conducted as a Skype Video Call. Started with an introduction, basic things I liked in my field, projects (just a gist). She shared a Google Doc link with me on Skype itself where I had to code. Then she quickly jumped onto the DSA problems.

Problem 1
Given an array of integers and an integer K, you need to find the total number of continuous subarrays whose sum equals K.

https://leetcode.com/problems/subarray-sum-equals-k/

This is a very well known interview problem for Hashmaps. Still she poked in a lot of times, asked multiple questions in between, and discussed the performance of Hashmaps, about expected constant time and randomness. It took quite some time to show why Hashmaps work in amortized constant time (she wanted me to explain it to her and I actually had a way to prove it). Also I talked about rehashing which is one bottleneck in the DS's performance, about bucketing, and about proper ways to use Hashmaps in C++ to avoid memory leaks and unnecessary bloating.

After all of this discussion I implemented the code in the most neatly indented, clean and efficient way possible, with all of the best coding practices (explaining each and every thing, about why I wrote it like this). Also I had to do a dry run, and explain the space and time complexities.

Then she asked me another variant of this problem, when K is fixed as 0. And she asked me how I can optimize the efficiency of my previous code. Also, this time I didn't have to count the total subarrays, I just had to tell whether there existed a subarray with sum 0 or not.

I replaced the Hashmap with a Hashset, and told her that it's the best we can do. She was again in a mood to debate over it, but now again I proved it to her that we can't do any better than this (very politely, never be rude in an interview!).

By this point I knew she is not at all friendly, and I can't know for sure if she is satisfied or not. But I hadn't left even a single bit unexplained (and with utmost politeness). I asked her if she was satisfied with my solutions, she replied yes (but her tone was flat throughout, maybe she was not at all expressive).

Problem 2
Again a standard one:

https://leetcode.com/problems/merge-k-sorted-lists/

I told her about the most efficient solution. Then she started asking about the min heap based data structure that I was using (priority_queue in C++) along with it's template arguments. She asked me a couple more of such questions.

Now, I coded the solution again in the most neatly indented, clean and efficient way possible, with all of the best coding practices (explaining each and every thing, about why I wrote it like this). Also I had to do a dry run, and explain the space and time complexities.

Then she asked me about a few variants of this problem, and the changes that needed to be made in the existing code, and how it will affect the efficiency, and what could be done to improve. I told her about everything she asked.

At this point she started sounding more impressed and satisfied.

Now we had one Object Oriented Design problem ahead.

Problem 3
An OOP design problem for “Deck of Cards”.

Creativity was all left to me. She gave me total freedom on how to design. First I created some layout and listed all properties I knew about cards, then segregated them into “basic” and “specific”. Now a few things were weird (unrealistic in some sense in regards to cards) I guess so I told her that I haven’t played cards much, just have solved probability and combinatorics problems on them, so pardon me.

To my surprise she laughed, it was a positive sign for me I guess. Anyways, I did strategize everything before jumping onto the code, which might have impressed her, and then I gave a very simple code design. And it was all done.

It was supposed to be a 45 minute round, but it went for an hour and 20 minutes.

At the end she asked me if I had anything to ask, I did ask a very sensible question about the gap she has observed in a recent grad and a professional, and what skill set does one need to have before joining. She told me that it’s a very good question, and was very happy answering it, she actually gave a long answer.

Advice

1) Stay calm and positive, especially with a neutral, unhelpful and rude interviewer.
2) Try to come up with ways of explaining things differently if one doesn't get your point at once.
3) If possible, come up with instant proofs, that will save time in arguments.
4) Satisfying the recruiter is the most important thing, and it is the key.

Now, surprisingly, I got a call directly to the final (third) round! No Technical Round 2 for me!

Technical Round 3

Again a Skype Video Call. Introduction first, then interests in the field, any experience I had and stuff. I told about my CP journey. And then he asked me to explain briefly about my projects. After all of this, he asked me to explain in detail about my favourite project. I talked about it in full detail for about 10 minutes. I covered it all, he didn’t have any questions.

Then he asked me about various OS problems like Deadlocks, Starvation, etc. and their prevention measures.

Along the same lines I told him about an OS project that I had done, the moment he heard about it, he said “I think everyone has done this”, then I said that this project was a part of a course work that I

took. Then I explained to him why my project is unique and how much up to date it is with the latest coding standards and kernel implementations. Also, I told him that I had first read the latest Linux kernel source to see how things work in the modern kernels, and then implemented something way different from what was mentioned in the decades old book. I also talked about other technicalities of this project in detail.

At this point he got very much impressed, and gave me very positive feedback. Also, he mentioned that he hadn't seen a similar project in the department, so it really is very unique.

Then a bit of design and Software Engineering, good practices, importance of collaborative work, working in teams.

Then a few behavioral questions like toughest challenges you've faced. Any issues you're still struggling with, any challenges with relocation, etc. I tried to maintain utmost professionalism here, and he was satisfied with my responses.

At the end he asked me if I had some questions for him. I asked about the difference between the mentality of a novice programmer and a professional, and what path a novice should take to become a professional in real sense.

He was very much impressed and he even mentioned that no one has ever asked him such a relevant question before and he talked for more than 15 minutes.

Again this round was supposed to be a 30 minute round and it went for about 50 minutes. He was very much impressed with all of my responses throughout the interview and gave me very positive feedback.

Advice

1) Being confident is the key.
2) Try to sound intellectual and professional, even the questions you ask them are being judged, so be very careful with whatever you speak.
3) Building a good rapport is always a positive sign and is very helpful.
4) Try to quantify things, like the difference you made, the performance increase a new technique can bring, the uniqueness (most important), etc.