Software Development Engineer Preparation Guide

Technical Interviews

CS Fundamentals: Demonstrate a strong understanding for CS/engineering fundamentals. Be sure to brush up on your data structures, algorithms, system design, and architecture knowledge. Think about how you would consider scalability, concurrency, and/or distributed systems in your solutions. Will your solution efficiently scale to tens or hundreds of millions of users? What are the bottlenecks and how could you remove them?

Coding:
Expect to write syntactically correct code— not just pseudo code or brute force solutions. If you feel a bit rusty coding without an IDE, it's a good idea to get comfortable coding with a pen and paper. While familiarity with any particular language is not required, it is important to be proficient in at least one language for writing code. The most important responsibility of a Software Development Engineer at Amazon is to write scalable, robust, and well-tested code. A few missed commas or typos are not a deal breaker, but the goal is to write code that is as close to production ready as possible. Make sure to check for edge cases and validate that no bad input can slip through. We want simple, optimal, and clear solutions.

Algorithms:
Your interviews will not be focused on rote memorization of algorithms; however, having a good understanding of common algorithms will help in solving some of the questions we ask. Consider reviewing traversals, searching, divide and conquer, and other common algorithms. Knowing the runtimes, theoretical limitations, and basic implementation strategies of different classes of algorithms is more important than memorizing the specific details of any given algorithm. When discussing algorithms problems, it is also important to show your work and thought process.

Data Structures:
Much of the work we do involves storing and accessing data efficiently. You will be expected to understand the inner workings of common data structures and be able to compare and contrast their usage in various applications. You will be expected to know the runtimes for common operations, as well as how they use memory.

Object-Oriented Design:
Good design is paramount to extensible, bug free, long-lived code. It is possible to solve any given software problem in almost a limitless number of ways, but when software needs to be extensible and maintainable, good software design is critical to success. Using object-oriented design best practices is one way to build lasting software. You should have a working knowledge of a few common and useful design patterns, as well as know how to write software in an object-oriented way with the appropriate use of inheritance and aggregation. You won't be tested on how specific design patterns work, but expect to be able to defend your design choices.

System Design, Operating Systems, and Databases:

- **System Design:** While we have some internal tools that help us with scaling, it is important to have an understanding of a few basic system design concepts. Examples include service oriented architecture, caching, and multi-tiered architecture.

- **Operating Systems:** You won't need to know how to build your own operating system from scratch, but you should be familiar with some topics that can impact code performance, such as memory management, synchronization, paging, and multithreading.
- **Databases:** Most of the software that we write is backed by a data store, somewhere. Many of the challenges we face arise when figuring out how to most efficiently retrieve or store data for future use. Amazon has been at the forefront of the non-relational database movement. We have made Amazon Web Services (i.e. DynamoDB) available for the developer community, which lets them easily leverage the benefits of non-relational databases. The more you know about how relational and non-relational databases work and what tradeoffs exist between them, the better prepared you will be.

Advanced Topics:

- **Distributed Computing:** Systems at Amazon have to work under very strict tolerances at a high load. Having an understanding of topics such as workflow systems, map-reduce, distributed caching schemes, and load balancing can help you formulate answers to some of the more complicated distributed architecture questions you might encounter.
- **Client Development:** Amazon's client applications also operate at an enterprise level. The development of these client applications can require familiarity with graphics libraries, computational geometry, hardware integration, content protection schemes, and System Development Kit (SDK) design.

Problem Solving: Do not make assumptions on how to solve the problem. We want to know how you deal with ambiguity and break complex problems into manageable parts. We are looking for candidates to show the natural ability and initiative to ask clarifying questions before they begin to solve the problem, which is important when working on intricate, large-scale technical projects. Thinking out loud during your interview is not required, but it will help the interviewers understand your thoughts, which will allow them to help you along the way. If you get stuck during your interviews, listen for and take advantage of the hints that the interviewers may provide to guide you to a successful solution.

Technical Resources to Study:

- http://www.careercup.com/ - Site dedicated to helping candidates prepare for technical interview questions. A section is dedicated to Amazon interviews.
- Cracking the Coding Interview: 150 Programming Questions and Solutions by Gayle Laakmann McDowell
- Introduction to Algorithms by Thomas H. Cormen, Charles E. Leiserson, et al.
- https://projecteuler.net/ - A series of increasingly difficult math/programming problems.

Leadership Principles: Amazon assesses all candidates on our Leadership Principles. Be ready to show how your experience and values overlap with these. While you do not have to memorize the leadership principles, please study them and be prepared to explain how you have embodied these principles in your own experiences. You will want to have 2-3 detailed examples for each principle. Try to avoid using the same examples throughout your interviews; it is important that we see the breadth of your work and results.

Other Tips:

- When answering questions, be as concise and detailed in your response as possible. The STAR (situation, task, action, result) method is recommended.
- We want to hear your answers in terms of "I" rather than "we" so that we understand your individual impact, versus that of your team
- Amazon is a very data and results driven company. We make decisions with data whenever possible. Each of your examples should include the data you used in your decision making and what the results were.
- Amazon values candidates who are self-aware of their failures. Do not be afraid to talk about your mistakes and how you've learned from them.
- Be ready to discuss the Amazon products and/or technologies that excite you. While we do not expect you to know every detail about the team you are interviewing with, you should research the team and product enough to be able to have a thorough conversation with the interviewers on those topics.
- You will have time during each interview to ask questions. Please be prepared with thoughtful questions to ask the interview team.