Assessment Prep

Online Technical Assessment

OVERVIEW

Why an assessment?
At Amazon, one of our highest priorities is hiring and developing the best. We work hard to raise the performance bar with every hire. Amazon uses online assessments as one way to help us get to know you better. We design them to measure key characteristics required for success in our Software Engineer roles.

Assessments also allow us to assess applicants consistently and equitably, as every individual is provided with the same experience and information needed to complete the assessment. Completing the assessment is an important first step in the recruiting process.

What does an SDE do at Amazon?
They tackle some of the most complex challenges in large-scale computing. They own the software development life cycle, design solutions, and work on coding, testing, implementing, maintaining and iterating solutions. With a strong understanding and applicable knowledge of CS fundamentals, things like algorithms and data structures are innate to them.

OBJECTIVE

Upon completion, your code assessment will be reviewed by a member of our engineering team.

We are interested in your:
- Demonstration of problem solving
- Writing correct code
- Applying patterns/data structures
- Optimizing for algorithmic performance

You can get a feel for the interface and functionality by completing this 60 minute (2 question) ASSESSMENT DEMO.

Please also review HackerRank test-taking tips prior to starting your assessment.

INSTRUCTIONS

Please ensure you have sufficient time to complete the test in one sitting before you begin. Once the timer starts, it will not stop until the coding assessment portion is complete. You will have 105 minutes to complete 2 coding challenges and coding approach. The time remaining will be clearly displayed in the left hand corner of the screen throughout the test. There will be 2 surveys following the assessment; a work style survey (15 min) and feedback survey (5 min).

You will have the option of coding in C, C++, C++14, C#, Go, Java7, Java8, Javascript, Kotlin, Objective-C, PyPy2, PyPy3, Python 2, Python 3, Ruby, Scala, and Swift. You may use different coding languages for the two questions; however, switching your language selection for a question that you have already started coding will delete all items you have already coded for that question.

Your code is being auto-saved periodically. You can also save by clicking on the SAVE button. In case of a system failure you can resume from the last saved instance. Your code will also auto-save when you click on NEXT QUESTION if you are moving back and forth between questions.

If you need to take a break, the timer will stop following the completed assessment. If you choose to take a break, log out before beginning the work style survey. When you're ready to start, log back in using the same credentials and you will return to where you left off last.

Efficiency and optimization, as opposed to brute force solutions, earn more points! Your code must compile for all code questions in order to move forward in the interview process. Be sure to test your code and ensure it runs before you submit your code or before time runs out.

You can compile your code as many times as you like during the assessment, but there must a 15 second interval between consecutive compilations.

Ensure your solutions consider all edge cases and handle large inputs effectively. This is key to doing well in the assessment.

NOTE

We do not penalize use of your own IDE while completing the code challenge; however, our engineers will manually review the assessment including the execution statistics to determine next steps in the recruiting process.

We highly recommend showing as much of your work in the online platform to provide our engineers insight into your problem solving approach.

You may use the resources that are available for all candidates (e.g. JDK or STL). Use the in-browser editor to work on the coding challenge and elaborate your coding approach as much as possible. Your coding and elaboration skills are both considered in our evaluation.

TIP

For the best experience you should use the latest stable versions of **Google Chrome, Mozilla Firefox, Apple Safari**, or **Microsoft Edge.**

The site will not function satisfactorily with IE11 or earlier.

LEARN MORE

Dive into our Leadership Principles
Learn more about Amazon
Explore Interviewing at Amazon for FAQs, prep guides and more

QUESTIONS? REACH OUT TO YOUR RECRUITING POINT OF CONTACT

AMAZON IS AN EQUAL OPPORTUNITY EMPLOYER

TECHNICAL TOPICS

Programming Languages
Familiarity with a prominent language is a prerequisite for success. Be familiar with the syntax and nuances of common languages – Java | Python | C# | C | C++ | Ruby | JavaScript.

Data Structures
Storing and providing access to data in efficient ways. Understand the inner workings of common data structures and be able to compare and contrast their usage in various applications. Know the runtimes for common operations as well as how they use memory. Wikipedia is a great resource for brushing up on data structures.

Algorithms
Basic implementation strategies of different classes of algorithms is more important than memorizing the specific details of any given algorithm. Consider reviewing traversals and divide and conquer algorithms. Consider knowing how and when to use a breadth-first search vs. a depth-first search, and what the trade-offs are. Knowing the runtimes, theoretical limitations, and basic implementation strategies of different classes of algorithms > memorizing specific details of any given algorithm.

Coding
The most important thing a Software Development Engineer does at Amazon is write scalable, robust, and well-tested code. Be comfortable coding by hand. Expect to be asked to write syntactically correct code - no pseudo code. Check for edge cases and validate that no bad input can slip through. The goal is to write code that's as close to production ready as possible. This is your chance to show off your coding ability.

Object-Oriented Design
Good design is paramount to extensible, bug free, long-lived code. Using object-oriented design best practices is one way to build lasting software. Have a working knowledge of a few common and useful design patterns. Know the appropriate use of inheritance and aggregation. Expect to defend and describe your design choices.

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. The more you know about how relational and non-relational databases work and what trade-offs exist between them, the better prepared you will be.

HELPFUL LINKS

CODE ASSESSMENT DEMO

HACKERRANK.COM - FAQS

WWW.CODECHEF.COM

PROJECTEULER.NET

CODE.GOOGLE.COM

LEETCODE.COM

HIGHSCALABILITY.COM

MYCODESCHOOL

YOU SHOULD HAVE

REVIEWED THIS GUIDE AND THE HACKERRANK FAQS IN THEIR ENTIRETY

COMPLETED THE ASSESSMENT DEMO

2.5 UNINTERRUPTED HOURS

STABLE INTERNET ACCESS

LAPTOP/COMPUTER

CHARGER

MAXIMUM 3 BROWSER SESSIONS RUNNING

QUESTIONS? REACH OUT TO YOUR RECRUITING POINT OF CONTACT