Preparing for your phone interview

Before you start preparing for your interview, it's good to have a general understanding of our interview process. We highly recommend that you do your research on interviewing at Google, and a great place to start is learning about how we hire. You can also check out Dean Jackson's ACM article on technical interviews at Google for great tips on the process and how to practice.

Interview topics may cover anything on your resume, especially if you have stated that you are an expert in a particular area. Be prepared to write **code on a shared Google doc**, build and develop complex algorithms, analyze their performance characteristics, data structures, and logic problems. Finally, know your core Computer Science principles: hash-tables, stacks, arrays, API's, Objected Oriented Design & Programming etc. Computer Science fundamentals are prerequisites for all engineering roles at Google, regardless of seniority, due to the complexities and global scale of the projects you would end up participating in.

You should focus on demonstrating your problem solving skills, applied to the question asked. If it's a coding question, providing efficient, prototype code in a short time frame to solve the problem is the key. We will also look for you to optimize your answer where possible. If it's a design question, working with your interviewer to create a high-level system, at times perhaps diving deeper on particular salient issues, is what matters. If it's a general analysis question, show that you understand all particularities of the problem described and (where applicable) offer multiple solutions, discussing their relative merits. Every Interviewer ultimately wants to answer the question of whether they'd be comfortable working with you on their team.

Sample Topics:

Coding: You should know at least one programming language very well, preferably C++ or Java. For specific projects, we do also use C and Python. You will be expected to write code in most of your interviews. You will be expected to know a fair amount of detail about your favorite programming language. Make sure to check out our Google code style guides. Your code should be well written, consistently styled, and as close to compilable as possible. You will be expected to know about API's, OOD/OOP, how to test your code, as well as come up with corner cases and edge cases for yours and other peoples code.

Algorithms: You should approach the problem with both from the bottom-up and the top-down. Algorithms that solve many Google problems are, Sorting (plus searching and binary search), Divide-and-Conquer, Dynamic Programming / Memorization, Greediness, Recursion or algorithms linked to a specific data structure. For more information on algorithms (and practice problems), you can visit TopCoder.

Data structures: You should study up on as many data structures as possible. Data structures most frequently used are Arrays, Stacks, Queues, Hash-sets, Hash-maps, Hash-tables, Dictionary, Trees and Binary Trees, Heaps, Graphs. You should know the data structure inside out, runtime complexity for most operations, and what algorithms tend to go along with each data structure.

Interview Tips

Substantiate - Make sure that you substantiate what your CV/resume says – for instance, if you list Java or Haskell as your key programming language, be prepared to be asked questions about it.

Explain - Explain your thought process and decision making throughout the interview. In all of Google's interviews, our engineers are evaluating not only your technical abilities but also how you approach problems and how you try to solve them. We want to understand how you think. This would include ***explicitly stating and checking any assumptions*** you make in your problem solving to ensure they are reasonable.

Clarify - Ask clarifying questions if you do not understand the problem or need more information. Many of the questions asked in Google interviews are deliberately underspecified because our engineers are looking to see how you engage the problem. In particular, they are looking to see which areas you see as the most important piece of the technological puzzle you've been presented.

Improve - Think about ways to improve the solution that you present. In many cases, the first solution that springs to mind isn't the most elegant and may need some refining. It's worthwhile to talk through your initial thoughts with the interviewer. Jumping immediately into presenting a brute force solution will be less well received than taking time to compose a more efficient solution.

Practice - Make sure you practice writing code on a Google doc. Be sure to test your own code and ensure it's easily readable without bugs.

Do some reading to prepare - In order to help you prepare for your interviews, the following literature has been recommended by Googlers:

Programming Interviews Exposed: Secrets to Landing Your Next Job, John Mongan, Eric Giguere, Noah Suojanen, Noah Kindler, John Wiley & Sons
Introduction to Algorithms, Thomas H. Cormen
Cracking the Coding Interview, Gayle Laakmann McDowell

Get some practice!
To practice for your interview, you may want to try:

Tech Interviews @ Google

Google Code Jam questions samples from Google coding competitions.

Take Dean's advice! Try practicing coding in a Google doc or on a whiteboard with a friend.