Google Software Engineer Interview Guide

- Curated by Aman Barnwal

Google coding interviews are really challenging. The questions are difficult, specific to Google, and cover a wide range of topics.

The good news is that the right preparation can make a big difference. We've analyzed 170+ software engineer interview questions reported by Google candidates, in order to determine the most frequently asked types of questions. Below, we've provided a curated list of real example questions, including free solutions.

In addition, you'll find preparation tips and links to the best resources, so that you can prepare more strategically and maximize your chances of landing that software engineer job at Google.

Here's an overview of what we'll cover:

Note that we have separate guides for Google engineering managers, data engineers, and machine learning software engineers, so take a look at those articles if they are more relevant to you.

---

1. Interview process and timeline

1.1 What interviews to expect

What's the Google software engineering interview process and timeline? It usually takes more than eight weeks and follows these steps:

1. Resume, cover letter, and referrals
2. Online assessment (new graduates and interns only)
3. Technical phone screen: one to two interviews
4. Onsite interviews: four to six interviews

Note that the exact process varies slightly from position to position. Your recruiter will send you a PDF at the beginning of the process which details what interviews you can expect. Here are a few example PDFs you might receive: software engineer, engineering manager, and front-end mobile engineer.

1.1.1 Google online assessment (90 minutes) - new graduates and interns only

If you're applying for a new graduate or intern position your process will often start with a coding sample test to take online. The coding sample includes two questions that you have to complete in less than 90 minutes in total.

The questions are similar to the ones you'll be asked in your interviews (i.e. data structures and algorithms). Note that you'll need to write your own test cases as you won't be provided with any. You can do that in your own IDE before submitting your solution. To pass to the next round you usually need to solve both of the questions correctly.

Check out the archives of coding competitions Google organizes to get an idea of the type of questions you'll come across. We recommend looking at the Code Jam competition in particular. Leetcode also maintains a thread on what questions to expect in Google's sample coding test. You can also find a list of preparation tips in our Google online assessment guide.

1.1.2 Technical phone screen

If you're an experienced hire, or if you are a new graduate who has passed the coding sample test, you'll be invited to one or two technical phone screens. This step is called the "phone screen", but most of the time it takes place over video chat on Google Hangouts / Google Meet. Each interview will last 30 to 60 minutes. You'll speak to a peer or a potential manager and they'll ask you to solve data structure and algorithm questions.

You'll share a Google Doc with your interviewer, write your solution directly in the document and won't have access to syntax highlighting or auto-completion like you would in a regular IDE. It's therefore a good idea to practice writing code in Google Docs before your interview.

Finally, in addition to coding questions, you should also be ready to answer a few typical behavioral questions including "Tell me about yourself," "Why Google?" or, "Tell me about a recent project you worked on." These questions are less frequent than they are in engineering interviews at Facebook or Amazon but it's still a good idea to think through the main ones ahead of time.

1.1.3 Onsite interviews

Onsite interviews are the real test. You'll typically spend a full day at a Google office and do four to six interviews in total. Each interview will last about 45 minutes and cover one of the following topics:

- Coding interviews
- System design interviews
- Leadership interviews (management positions only)

You'll typically get three coding interviews with data structure and algorithm questions, and one or two system design interviews. All candidates are expected to do extremely well in coding interviews. If you're relatively junior (L4 or below) then the bar will be lower in your system design interviews than for mid-level or senior engineers (e.g. L5 or above).

You'll use a whiteboard to write your code in most onsite interviews at Google. But the company has also started offering Chromebooks for coding interviews at some locations. These laptops come with an interview app that lets you choose the coding language you want to use.

In addition, if you're applying for a management position (e.g. Engineering Manager) then you'll also have leadership interviews where you'll be asked behavioral questions about leading teams and projects.

Finally, in addition to interviews, you'll also have lunch with a fellow engineer while you are onsite. The lunch interview is meant to be your time to ask questions about what it's like to work at Google. The company won't be evaluating you during this time, but we recommend that you behave as if they are.

1.2 What exactly is Google looking for?

At the end of each interview your interviewer will grade your performance using a standardised feedback form that summarizes the attributes Google looks for in a candidate. That form is constantly evolving, but we have listed the main components we know of at the time of writing this article below.

A) Questions asked

In the first section of the form the interviewer fills in the questions they asked you. These questions are then shared with your future interviewers so you don't get asked the same questions twice.

B) Attribute scoring

Each interviewer will assess you on the four main attributes Google looks for when hiring:

1. **General cognitive ability.** This is often referred to as "GCA" by Googlers. The company wants to hire smart engineers who can learn and adapt to new situations. Here your interviewer will try to understand how you solve hard problems and how you learn. For more information, take a look at our guide to the GCA interview.
2. **Role-related knowledge and experience.** This is often referred to as "RRK" or "RRKE" internally. The company wants to make sure that you have the right experience, domain expertise and competencies for the position you're applying for. For more information, take a look at our guide to the RRK interview.
3. **Leadership.** Google looks for a particular type of leadership called “emergent leadership.” You'll typically be working in cross-functional teams at Google, and different team members are expected to step up and lead at different times in the lifecycle of a project when their skills are needed.
4. **Googleyness (i.e. culture fit).** The company wants to make sure Google is the right environment for you. Your interviewer will check whether you naturally exhibit the company's values including: being comfortable with ambiguity, having a bias to action, and a collaborative nature.

Depending on the exact job you're applying for these attributes might be broken down further. For instance, "Role-related knowledge and experience" could be broken down into "Security architecture" or "Incident response" for a site reliability engineer role. But the total number of attributes does not usually exceed six or seven.

In this middle section, Google's interviewers typically repeat the questions they asked you, document your answers in detail, and give you a score for each attribute (e.g. "Poor", "Mixed", "Good", "Excellent").

C) Final recommendation

Finally interviewers will write a summary of your performance and provide an overall recommendation on whether they think Google should be hiring you or not (e.g. "Strong no hire", "No hire", "Leaning no hire", "Leaning hire", "Hire", "Strong hire").

1.3 What happens behind the scenes

If things go well at your onsite interviews here is what the final steps of the process look like:

- Interviewers submit feedback
- Hiring committee recommendation
- Team matching
- Senior leader and Compensation committee review
- Final executive review (only senior roles)
- You get an offer

After your onsite, your interviewers will all submit their feedback usually within two to three days. This feedback will then be reviewed by a hiring committee, along with your resume, internal referrals, and any past work you have submitted. At this stage, the hiring committee will make a recommendation on whether Google should hire you or not.

If the hiring committee recommends that you get hired you'll usually start your team matching process. In other words, you'll talk to hiring managers and one or several of them will need to be willing to take you in their team in order for you to get an offer from the company.

In parallel, the hiring committee recommendation will be reviewed and validated by a Senior manager and a compensation committee who will decide how much money you are offered. Finally, if you are interviewing for a senior role, a Senior Google executive will review a summary of your candidacy and compensation before the offer is sent to you.

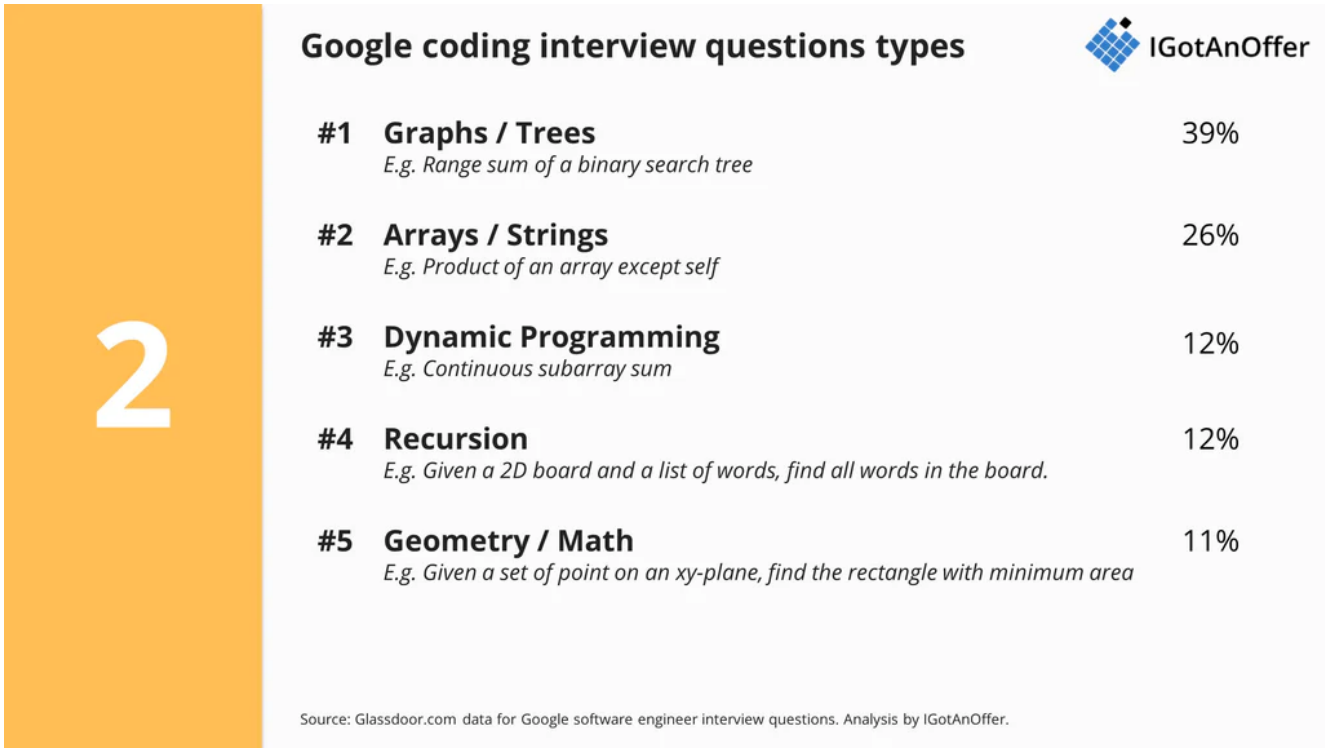
As you've probably gathered by now, Google goes to great lengths to avoid hiring the wrong candidates. This hiring process with multiple levels of validations helps them scale their teams while maintaining a high caliber of employees. But it also means that the typical process can spread over multiple months.

2. Example questions

We believe in data-driven interview preparation and have used Glassdoor data to identify the types of questions which are most frequently asked at Google.

For coding interviews, we've broken down the questions you'll be asked by subcategories (e.g. Arrays / Strings , Graphs / Trees , etc.) so that you can prioritize what to study and practice first. For system design and behavioral questions, we've listed questions that were frequently reported on Glassdoor and other resources.

2.1 Coding & Algorithm interview



Google software engineers solve some of the most difficult problems the company faces with code. It's therefore essential that they have strong problem solving skills. This is the part of the interview where you want to show that you think in a structured way and write code that's accurate, bug-free, and fast.

Here are the most common question types asked in Google coding interviews and their frequency. Please note the list below excludes system design and behavioral questions, which we'll cover later in this article.

1. Graphs / Trees (39% of questions, most frequent)
2. Arrays / Strings (26%)
3. Dynamic programming (12%)
4. Recursion (12%)
5. Geometry / Maths (11% of questions, least frequent)

Below, we've listed common examples used at Google for each of these different question types. To make these questions easier to study, we've modified the phrasing to match the closest problem on Leetcode or another resource, and we've linked to a free solution.

Finally, we recommend reading our guide on how to answer coding interview questions to understand more about the step-by-step approach you should use to solve these questions.

Example coding questions asked at Google

1. Graphs / Trees (39% of questions, most frequent)

- "Given a binary tree, find the maximum path sum. The path may start and end at any node in the tree." (Solution)
- "Given an encoded string, return its decoded string." (Solution)
- "We can rotate digits by 180 degrees to form new digits. When 0, 1, 6, 8, 9 are rotated 180 degrees, they become 0, 1, 9, 8, 6 respectively. When 2, 3, 4, 5, and 7 are rotated 180 degrees, they become invalid. A *confusing number* is a number that when rotated 180 degrees becomes a different number with each digit valid. (Note that the rotated number can be greater than the original number.) Given a positive integer *N*, return the number of confusing numbers between *1* and *N* inclusive." (Solution)
- "Given two words (*beginWord* and *endWord*), and a dictionary's word list, find the length of shortest transformation sequence from *beginWord* to *endWord*, such that: 1) Only one letter can be changed at a time and, 2) Each transformed word must exist in the word list." (Solution)
- "Given a matrix of N rows and M columns. From m[i][j], we can move to m[i+1][j], if m[i+1][j] > m[i][j], or can move to m[i][j+1] if m[i][j+1] > m[i][j]. The task is print longest path length if we start from (0, 0)." (Solution)
- "Given a robot cleaner in a room modeled as a grid. Each cell in the grid can be empty or blocked. The robot cleaner with 4 given APIs can move forward, turn left or turn right. Each turn it made is 90 degrees. When it tries to move into a blocked cell, its bumper sensor detects the obstacle and it stays on the current cell. Design an algorithm to clean the entire room using only the 4 given APIs shown below." (Solution)

2. Arrays / Strings (26%)

- Implement a SnapshotArray that supports pre-defined interfaces (note: see link for more details). (Solution)

- "In a row of dominoes, `A[i]` and `B[i]` represent the top and bottom halves of the `i`-th domino. (A domino is a tile with two numbers from 1 to 6 - one on each half of the tile.) We may rotate the `i`-th domino, so that `A[i]` and `B[i]` swap values. Return the minimum number of rotations so that all the values in `A` are the same, or all the values in `B` are the same. If it cannot be done, return `-1`." (Solution)
- "Your friend is typing his *`name`* into a keyboard. Sometimes, when typing a character *c*, the key might get *long pressed*, and the character will be typed 1 or more times. You examine the *`typed`* characters of the keyboard. Return *`True`* if it is possible that it was your friends name, with some characters (possibly none) being long pressed." (Solution)
- "Given a string S and a string T, find the minimum window in S which will contain all the characters in T in complexity O(n)." (Solution)
- "Given a list of query words, return the number of words that are stretchy." Note: see link for more details. (Solution)
- "Given an array of words and a width *maxWidth*, format the text such that each line has exactly *maxWidth* characters and is fully (left and right) justified." (Solution)

3. Dynamic Programming (12%)

- "Given a *`matrix`* and a *`target`*, return the number of non-empty submatrices that sum to `target`." (Solution)
- "Given a *rows x cols* binary *matrix* filled with *0's* and *1's*, find the largest rectangle containing only *1's* and return its area." (Solution)
- "Your car starts at position 0 and speed +1 on an infinite number line. (Your car can go into negative positions.) Your car drives automatically according to a sequence of instructions A (accelerate) and R (reverse)...Now for some target position, say the length of the shortest sequence of instructions to get there." (Solution)
- "Given strings `S` and `T`, find the minimum (contiguous) substring `W` of `S`, so that `T` is a subsequence of `W`. If there is no such window in `S` that covers all characters in `T`, return the empty string `""`. If there are multiple such minimum-length windows, return the one with the left-most starting index." (Solution)

4. Recursion (12%)

- "A strobogrammatic number is a number that looks the same when rotated 180 degrees (looked at upside down). Find all strobogrammatic numbers that are of length = n." (Solution)
- "Given a binary tree, find the length of the longest path where each node in the path has the same value. This path may or may not pass through the root. The

length of path between two nodes is represented by the number of edges between them." (Solution)

- "Given the `root` node of a binary search tree, return the sum of values of all nodes with value between `L` and `R` (inclusive). The binary search tree is guaranteed to have unique values." (Solution)

5. Geometry / Math (11% of questions, least frequent)

- "A group of two or more people wants to meet and minimize the total travel distance. You are given a 2D grid of values 0 or 1, where each 1 marks the home of someone in the group. The distance is calculated using Manhattan Distance, where distance(p1, p2) = `|p2.x - p1.x| + |p2.y - p1.y|`." (Solution)
- "You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order and each of their nodes contain a single digit. Add the two numbers and return it as a linked list." (Solution)

2.2 System design interview

Google Search, GMail, Google Docs, Android, and YouTube all have 1bn+ monthly active users. Google engineers therefore need to be able to design systems that are highly scalable and performant. The coding questions we've covered above usually have a single optimal solution. But the system design questions you'll be asked are typically more open-ended and feel more like a conversation.

This is the part of the interview where you want to show that you can both be creative and structured at the same time. In most cases, your interviewer will adapt the question to your background. For instance, if you've worked on an API product they'll ask you to design an API. But that won't always be the case so you should be ready to design any type of product or system at a high level.

As mentioned previously, if you're a junior developer the expectations will be lower for you than if you're mid-level or senior. In addition, for certain roles (e.g. infrastructure, security, etc.) you will likely have several system design interviews instead of just one.

Here are the most common system design questions asked in the Google interview reports which can be found on Glassdoor. For more information, we recommend reading this guide and practicing system design questions using our list of 31 questions.

Example system design questions asked at Google

- How would you design Google's database for web indexing

- How would you design Google Docs
- How would you design Google Home (voice assistant)
- How would you design Amazon's books preview
- How would you design a social network
- How would you design a task scheduling system
- How would you design a ticketing platform
- How would you design a system that counts the number of clicks on YouTube videos
- How would you design a webpage that can show the status of 10M+ users including: name, photo, badge and points
- How would you design a function that schedules jobs on a rack of machines knowing that each job requires a certain amount of CPU & RAM, and each machine has different amounts of CPU & RAM? Multiple jobs can be scheduled on the same machine as long as it can support it

2.3 Leadership interview

All candidates should be prepared to answer standard behavioral questions in their interviews (e.g. "Tell me about yourself", "Why Google", etc.) These questions tend to be asked as an ice-breaker at the beginning of coding and system design interviews.

In addition, if you're interviewing for management or senior positions (e.g. engineering manager) you'll also usually have leadership interviews where you'll be assessed on your ability to lead people and projects.

People management interviews tend to dive into how you would support and grow your team. You should expect questions about your approach to developing and retaining team members, your ability to lead teams through difficult situations, and why you think Google is the right culture for you.

Project management interviews tend to dive into how you would effectively lead projects end-to-end. You should expect questions about your overall project management philosophy, your ability to deal with complex and ambiguous situations, and your experience delivering results.

You'll be asked a mix of behavioral questions and hypothetical questions. Behavioral questions are about how you handled certain circumstances in the past. For instance, "Tell me about a time you lead a team through a difficult situation" is a behavioral

question. Hypothetical questions are about how you would handle a hypothetical situation. For instance, "How would you build a diverse and inclusive team" is a hypothetical question.

We've listed five standard non-coding interview questions that Google tends to ask to all software engineers below, and an additional 20 you should prepare for if you're targeting a management or senior position. For more information, check out our article on how to answer behavioral interview questions and the "Why Google?" question.

Example behavioral questions asked at Google

For all software engineers

- Tell me about yourself
- Why Google?
- Tell me about a recent / interesting project you worked on
- Tell me about a time you had to resolve a conflict in a team
- What is your favorite Google product

For management and leadership positions

People management interviews

- Tell me about a time you had to handle a project that was late
- Tell me about a time you had to handle trade offs and ambiguity
- Tell me about a time you were part of an organization in transition and how you helped them move forward
- Tell me about a time you lead a team through a difficult situation
- Tell me about a time you developed and retained team members
- How would you deal with a team challenge in a balanced way
- How would you address a skill gap or personality conflict
- How would you ensure your team is diverse and inclusive
- How would you organize day-to-day activities
- How would you convince a team to adopt new technologies

Project management interviews

- Tell me about a time you demonstrated leadership even though you weren't the formal manager

- Tell me about a time you were the end-to-end owner of a project
- Tell me about a time you used data to make a critical decision
- Tell me about a time you used data to measure impact
- How would you handle competing visions on how to deliver a project
- How would you choose a methodology to manage a project
- How would you balance flexibility and process in an agile environment
- How would you handle projects without defined end dates
- How would you prioritize projects of varying complexity
- How would you balance process vs. execution in an agile environment

3. How to prepare

Now that you know what questions to expect, let's focus on how to prepare. Here are the four most important things you can do to prepare for Google's software engineer interviews.

3.1 Learn about Google's culture

Most candidates fail to do this. But before investing tens of hours preparing for an interview at Google, you should take some time to make sure it's actually the right company for you.

Google is prestigious and it's therefore tempting to assume that you should apply, without considering things more carefully. But, it's important to remember that the prestige of a job (by itself) won't make you happy in your day-to-day work. It's the type of work and the people you work with that will.

If you know engineers who work at Google or used to work there it's a good idea to talk to them to understand what the culture is like. In addition, we would recommend reading the following resources:

- Google's mission statement (by Google)
- Google's values (by Google)
- Google strategy teardown (by CBS Insights)

3.2 Practice by yourself

As mentioned above, you'll have to answer three types of questions at Google: coding, system design, and behavioral. The first step of your preparation should be to brush up on these different types of questions and to practice answering them by yourself.

For coding interviews, we recommend getting used to the step-by-step approach hinted at by Google in the video below.

Here's a summary of the approach:

1. Ask clarification questions to make sure you understand the problem correctly
2. Discuss any assumptions you're planning to make to solve the problem
3. Analyse various solutions and tradeoffs before starting to code
4. Plan and implement your solution
5. Test your solution, including corner and edge cases

To practice solving questions we recommend using our articles, 73 data structure questions and 71 algorithms questions, which have links to high quality answers to each problem. They are organized by type of data structure or algorithm as well as by difficulty level. Don't forget to practice on a whiteboard or Google Doc instead of in an editor.

For the rest of your coding preparation, we recommend using our coding interview prep article as your one-stop-shop. It has a 7-step preparation plan and links to the best resources.

For system design interviews, we recommend studying our system design interview guide and learning how to answer system design interview questions. These guides cover a step-by-step method for answering system design questions, and they provide several example questions with solutions.

For behavioral interviews, we recommend learning our step-by-step method for answering this type of question. In addition, you'll want to write down your answers to the common behavioral questions we have listed in the previous section.

Finally, a great way to improve your communication for coding, system design, and behavioral questions, is to interview yourself out loud. This may sound strange, but it can significantly improve the way you communicate your answers during an interview. Play the role of both the candidate and the interviewer, asking questions and answering them out loud.

3.3 Practice with peers

Practicing by yourself will only take you so far. One of the main challenges of coding interviews is that you have to communicate what you are doing as you are doing it. To get used to this kind of "thinking out loud" we strongly recommend practicing live coding interviews with a peer interviewing you.

A great place to start is to practice with friends. If you don't have anyone in your network who can interview you, then you can also find peers to practice with on our mock interview platform.

3.4 Practice with ex-interviewers

Practicing with peers can be a great help, and it's usually free. But at some point, you'll start noticing that the feedback you are getting from peers isn't helping you that much anymore. Once you reach that stage, we recommend practicing with ex-interviewers from top tech companies.

If you know a software engineer who has experience running interviews at Google or another big tech company, then that's fantastic. But for most of us, it's tough to find the right connections to make this happen. And it might also be difficult to practice multiple hours with that person unless you know them really well.