

CS 61C: Great Ideas in Computer Architecture (Machine Structures) Career Advice

Instructors:
 Krste Asanovic
 Randy H. Katz

<http://inst.eecs.Berkeley.edu/~cs61c/F12>

11/26/12 Fall 2012 -- Lecture #X 1

How to Get a Job at Google

Google Jobs

About Google | Jobs | Life at Google | How we hire

How we hire

We're looking for our next Noogler - someone who's good for the role, good for Google and good at lots of things. Things move quickly around here. At Internet speed. That means we have to be nimble, both in how we work and how we hire. We look for people who are great at lots of things, love big challenges and welcome big changes. We can't have too many specialists in just one particular area. We're looking for people who are good for Google—and not just for right now, but for the long term.

This is the core of how we hire. Our process is pretty basic; the path to getting hired usually involves a first conversation with a recruiter, a phone interview and an onsite interview at one of our offices. But there are a few things we've baked in along the way that make getting hired at Google a little different.

How we interview

We're looking for smart, team-oriented people who can get things done. When you interview at Google, you'll likely interview with four or five Googlers. They're looking for four things:

| | | | |
|---|---|---|--|
| <p>Leadership</p> <p>We'll want to know how you've flexed different muscles in different situations in order to mobilize a team. This might be by asserting a leadership role at work or with an organization, or by helping a team succeed when you weren't officially appointed as the leader.</p> | <p>Role-Related Knowledge</p> <p>We're looking for people who have a variety of strengths and passions, not just isolated skill sets. We also want to make sure that you have the experience and the background that will set you up for success in your role. For engineering candidates in particular, we'll be looking to check</p> | <p>How You Think</p> <p>We're less concerned about grades and transcripts and more interested in how you think. We're likely to ask you some role-related questions that provide insight into how you solve problems. Show us how you would tackle the problem presented--don't get hung up on nailing the "right"</p> | <p>Googlyness</p> <p>We want to get a feel for what makes you, well, you. We also want to make sure this is a place you'll thrive, so we'll be looking for signs around your comfort with ambiguity, your bias to action and your collaborative nature.</p> |
|---|---|---|--|

11/26/12 Fall 2012 -- Lecture #X 4

1. Leadership

“We’ll want to know how you’ve flexed different muscles in different situations in order to mobilize a team. This might be by asserting a leadership role at work or with an organization, or by **helping a team succeed when you weren’t officially appointed as the leader.**”

Managing without authority

11/26/12 Fall 2012 -- Lecture #X 3

2. Role-Related Knowledge

“We’re looking for people who have a variety of strengths and passions, not just isolated skill sets. We also want to make sure that you have the experience and the background that will set you up for success in your role. For engineering candidates in particular, **we’ll be looking to check out your coding skills and technical areas of expertise.**”

What you know is what counts

11/26/12 Fall 2012 -- Lecture #X 4

3. How You Think

“We’re less concerned about grades and transcripts and more interested in how you think. We’re likely to ask you some role-related questions that provide insight into how you solve problems. Show us how you would tackle the problem presented--don’t get hung up on nailing the “right” answer.”

Problem-solving skills (e.g., <http://www.glassdoor.com/Interview/Google-Interview-Questions-E9079.htm>)

11/26/12 Fall 2012 -- Lecture #X 5

4. “Googlyness”

“We want to get a feel for what makes you, well, you. We also want to make sure this is a place you’ll thrive, so we’ll be looking for signs around your **comfort with ambiguity**, your bias to **action** and your **collaborative nature.**”

Working in a loosely structured environment

11/26/12 Fall 2012 -- Lecture #X 6

Getting A Job At Facebook: Inside The 'Meritocratic' Hiring Process

By Claire Gordon
Posted Oct 5th 2012 @ 7:59AM

11

Actionable Career Advice

- Take courses with *great* projects
 - Learn new technology (e.g., *Hadoop, EC2*) and develop new skills (e.g., *programming for performance*); Beyond book knowledge
- Have great industry internships
 - Exposure to real-world development environments: Test and documentation for stuff you leave behind
- Contribute to open source
 - Read, extend, and contribute to other people's code
- Build your tech portfolio
 - Write your own code!
 - Get to know and love github and git

11/26/12 Fall 2012 -- Lecture #X 8

Comments from Book Friends

Preparing an end-of-semester career advice lecture for sophomore computer science majors and wanna-bees. What are your best advice ideas? Serious answers only please!

Like Comment Share

View all 24 comments

David Oppenheimer BTW Rick, those were great advice! Friday at 22:26 · Like

Rick McGeer You too, David. Yesterday at 07:23 · Like

Louisa Raschid I would second the github account idea and also strongly encourage students to become involved / make contributions to a FOSS project. There are excellent lessons to be learned about "real" version control, long term maintenance, compatibility, standards, extensibility, etc. that cannot be addressed in a classroom situation. 16 hours ago · Like · +1

Rick McGeer Louisa, great advice. Matt Welsh first rose to stardom working on the Linux project in the early days, while he was an undergrad at Cornell. Working on a FOSS project with others also gives you access to great mentoring that is hard to get otherwise... See More

Seattle seattle.c.washington.edu Seattle makes the power of Cloud Computing available to everyone. It's free... See more

about an hour ago · Like · Remove Preview

Randy Howard Katz The overarching actionable advice is hack, build your own software, work on other people's software via participation in open source. All good! 16 minutes ago · Like

– Lecture #X 9

Comments from My Facebook Friends

- Rick McGeer, Senior Researcher, HP Labs
 1. Build it for yourself first, or for people around you.
 2. If you're doing software, study the work of a master craftsman, and pay attention to the lessons he leaves, both small and large.
 3. 90% of Performance Improvement is getting it up and running!
 4. Extensible, extensible, extensible: any system worth having outlasts its original use.
 5. Scope, scope, scope: build things that do a narrow range of things well, avoid doing too much.

11/26/12 Fall 2012 -- Lecture #X 10

Thread on Quora

Computer Science: Algorithms Computer Programming Electrical Engineering Higher Education Operating Systems Programming Languages Software Engineering Studies and Studying

Follow Question Promote Question

What advice would you give to a Computer Science major student that you wish you were given when you started Computer Science?

All advice is welcome! Edit

4 Comments · Share (21) · Options

60 Answers

Amy Quilpa, NYer studying CS at CMU 224 votes by Philip Mori, Jorge Ochoa, Allison Ho, (more)

Don't let other people intimidate you just because they've been doing this since they were six. Don't let other people make you feel like an idiot because you don't know something. Don't let other people convince you that your interests aren't good enough.

Listen to what other people have to say, and have an open mind, but don't let it get to you.

I know I'm answering this question differently than everyone else, but this was the kind of advice I really needed.

11/26/12 Fall 2012 -- Lecture #X 11

Thread on Quora

- Do these things between summers :
 - Focus on course work.
 - Study lot of discrete maths, linear algebra, algorithms & some machine learning
 - Do lot of programming competitions
 - Try to master at least two programming languages, preferably one very low level language like C, one relatively high level language like Python, and get familiar with non-imperative paradigms of programming. More the merrier.
 - Use Linux, and learn to use shell effectively
 - Understand how internet works, learn a few basic web development technologies & using them design your own home page
 - Stay updated with computer science world - both academia & industry

11/26/12 Fall 2012 -- Lecture #X 12

Thread on Quora

0000 – Buy your own domain name.
 0001 – Install an Apache web server and configure it in a non-trivial way, e.g. to serve multiple domains.
 0010 – Install WordPress and have your own blog. Write blog posts regularly. Write well. Good writing is a critical skill to master in this profession.
 0011 – Run your own web site at home or in a hosting company.
 0100 – Write at least one complete LAMP web app, preferably two — one where P=PHP, the other where P=Python.
 0101 – Have your own [physical or virtual] server on the cloud.
 0110 – Install VMWare or equivalent in order to boot up your laptop with more than one OS.
 0111 – Configure your home DSL router so that you serve a web site or other kind of server from your home machine / laptop to your friends.

11/26/12

Fall 2012 -- Lecture #X

13

Thread on Quora

1000 - Use a packet sniffer to learn about the network requests your computer does to your favorite game server.
 1001 - Make contributions to an open source project.
 1010 - Write an app that uses at least one of the popular Web APIs, like Facebook Connect or one of Google's.
 1011 - Use Google AdSense on your web site, and make money just by virtue of attracting traffic.
 1100 - Compile a complicated open source project from scratch, like OpenSim or Matterhorn.
 1101 - Read works of literature and, besides enjoying the ride, pay close attention to how the author tells the story and makes use of words. Your programs should be as carefully written as those works of art!
 1110 - Get yourself involved in a software project where requirements are bound to change halfway through — that's about 0.01% of homework projects and about 99.99% of real world projects, so find one of the latter kind. Finish the project with patience and the ability to take criticism in a constructive way.
 1111 - Write an application using map-reduce. Run it on Google app-engine or amazon EC2

11/26/12

Fall 2012 -- Lecture #X

14