Announcements

My Youtube Video Explaining the Process / Quick Start Guide :) Explains a bit about my story, process, resources, tip during the interview, etc. :) ^^ Hope helps ~ and be cool if u can support me ^^;;
https://www.youtube.com/watch?v=m71kZqOj5VU&feature=youtu.be

Calendly link for mock interviews / questions / resume reviews for free! Is inside the linktree link! Pls only keep one active booking at a time or I will cancel all your sessions.

This is more geared towards students b/c I am a student, rising senior, but if you feel like I can help you through a mock interview or basic Q/A / my experience feel free to book. I realize that during this pandemic it can be a scary / hard time so really just trying to give back however I can!
https://linktr.ee/justinwlin

Who Am I / Why am I writing this:

My name is Justin, I got a lot of help from people online when I was getting an internship and thought I would provide guidance to others.
(https://www.linkedin.com/in/justinlinw/)

This guide is **aimed mainly towards students** but might be good information for other ppl too. At least I hope so…

My experience was that in the summer of Freshmen year, I felt frustrated, so I taught myself full-stack web development online. From there, I worked hard, and luckily got into JP Morgan (which you can read more about here): (Document covers how I got into Amazon + JP Morgan)
SAT Reference Sheet (AMAZON/JP Morgan)

The gist is I got lucky, but I learnt a lot, and figured out "hey, I want to get into a FAANG company", and so I went through dozens of resources and got into Amazon! But yeah.. I also had to do a lot of trial and errors and I didn't understand what was going on… so I thought I would help future people out.

First watch this video.
(How to get into the Big 4)
https://youtu.be/YJZCUhxNCv8

This is a great video explaining and giving some motivation to study. Also will answer the questions, what should I put on my resume, how do I get into Google, Fang, etc, I mean the tldr is pass the interview, but it still might not have sunk in for a lot of people who worry that it is impossible that without any experience I could get into Google or FB etc.

But you can, and also you should be aiming towards these FAANG level companies.

Just think about it, check on levels.fyi, and you will see that a FAANG employee will earn around 150K starting new grad/junior dev total compensation. FAANG companies' interviews are also insanely standardized. They aren't asking you to be an amazing engineer, they are asking you to have good fundamentals and problem solving skills, and like learning the SAT you can study these things.

Look, getting into a large company, they are willing to train you. Working at JP Morgan, I had mentors that were spanish majors originally. Working at these large companies, they will invest time and resources b/c they have the ability to, they don't care what you know, b/c they will teach you and you will learn. All they ask is that you learn quickly and that you have something to contribute even if it is just your enthusiasm to learn and work.

However, let's say you try to apply to a smaller company, not saying it is bad, but as a new grad/junior I think there are better options b/c at these smaller companies they usually require:

- More knowledge
- Have less resources to train you
- Might have worst work-life balance (not always true, depends on the company)
- Get paid less

And it isn't the small companies' fault, it's just the reality of the situation that they don't have as much resources. So your goal, again, should be to get into FANG, or a similarly level company.

P.S Everyone always says: I send out 100 resumes and get no responses. Try to just limit / track it to FAANG companies when recruiting season is happening and you will probably find better response rates. FAANG companies have much more dedicated resources to process these resumes / interviews versus small companies that will lag behind or just never respond.

PLS PLS PLS CHECK OUT THE RESOURCES:
I RECOMMEND READING THE 50 PAGE GUIDE BY EDBERT CHAN and getting Grokking the Coding interview. Oh and obviously watching the youtube video.

Do take what Edbert says with a grain of salt, haha, but it is pretty accurate overall, and maybe a bit more focused on higher level engineers (L5/L6), if that is what you are looking for. But, while everything in this guide is something I think is important:

- Edbert's guide
- Grokking the Coding Interview
- Youtube video how how to get into the big 4

Are pretty much what I would tell people.

Also what language to use! Python!!! it is faster, easier to understand, and close to pseudocode. It is not hard to learn and most explanations are done in Java or Python online.

Python though is much less syntax heavy and allows for heavier abstraction and fits an interview where the point is analysis, explanation, and speed. Also the speed at which you can write python is so important bc leaves time to think, review, check over your work when things are more abstracted away. Ultimately, do what you want, but I just always recommend python.

Resume

First things first, if your resume is two columns, if it has graphics, if it has a picture, if you did it yourself in word and styled it, it probably looks bad. Your bullet points are probably bad if it is one sentence, or if you have more than 3 or 4 bullet points per experience you are probably too wordy. If you are greater than 1 page, cut it down (your resume is the summary of the most relevant things right now, NOT EVERYTHING, find this is a common problem with Masters and PHD students). Oh also, no jargon like: worked on blah blah driver in order to do cross analysis on xyz, blah blah. Like, realize a lot of people reading this might not have your knowledge and that HR people also do read a lot of these resumes before they get passed up, so you need to make it digestible to read. These are the most common issues.

Now, what do I recommend in terms of formatting:

- You can use Latex, can use sites like Overleaf to work with latex online and use nice resume templates.
- Keep it traditional and easy to read. Your most important resume experiences at the top.
- If you don't have any experience, take FREAKING UDEMY COURSES. They are like 10 bucks a course, no excuses. I started off with getting an internship at JP Morgan to Amazon, not b/c I had experience but b/c I studied my butt off on Udemy. Recommend Andrei Neagoie on Udemy, his courses with Complete Web Dev 2020 is probably the best if you want to go down the full-stack web dev route, or you can look at whatever specific courses you want.
- Recently was recommended by someone of this website (never tried it myself):
    - https://www.rezi.io/
    - And I think honestly is a decent option to get something that looks simple, traditional, clean to read.
- http://latexresu.me/
    - Another great resume builder someone recommended to me!
- https://2by22.blog/overhaul-resume-highly-effective-tips/?fbclid=IwAR27MQ5oQ7R4v0RiZj3ldHva5dWqMcpLCkAiIW6tVKJ0MBkx8_aJL9tWSQE

    - Take this link with a grain of salt, b/c developer resumes are written a bit differently, but this link was an inspiration to how I wrote my resume.
        - https://drive.google.com/file/d/12KYw0IbxWK0mYh4z-XP-dT5w5617pfbQ/view

Side addition (someone did note you can have nice looking resumes two columns, colors, etc). While true, I find most people are unable to accomplish this and use fancy resume templates they find online with meaningless bars and charts, too much wasted space, and so on.

Also two columns makes it harder for automatic parsers to read, which might or might not affect you. So while yes, you can have a great resume, just a standard template is also great, lower effort, and still gets the same practical benefit.

In terms of bullet-points:
Do not write!!! I worked at a company with 1 million dollar asset. I organized….

Write it neatly and concisely on **WHAT YOU DID, not what the company was or boring things like, I can use microsoft word ._.**

People can tell you are desperate at that point.
Example:
I worked on a react project to design a dashboard.

Can be improved by writing...
What did you do, why did you do it, if there are metrics you can qualify or quantitate it, or give it substance that you know what you are talking about.

Example reworked:
I developed a UI dashboard to help parse incoming data streams from our backend server, making it easier for clients to get a high level intuitive overview of what is going on. I was able to quickly pick up on React within only two weeks worth of time, create mock-ups, and deliver UI and functionality.

Here is my own resume:
https://drive.google.com/file/d/12KYw0IbxWK0mYh4z-XP-dT5w5617pfbQ/view

You will notice in my own resume, I like to add an additional bullet point at the end of every experience stating:
Technology: React, Redux, etc etc…

My resume isn't the best, but it will give an idea of how it should look.

What do companies look for on the resume?

Generally, from my experience, some sort of full-stack understanding or project that shows the knowledge that you say you have chosen to specialize into.

Ex. Questions:

- What did you do with this project?
- Why did you do it?
- How did you do it?

Ex. of detailed questions I got ask on my projects:

- Do you know what an API is?
- What is an HTTPS request
- How do you connect a frontend to a backend
- Do you know SQL versus noSQL
- What is the difference between a frontend and backend
- What is Test driven development?
- What is CI/CD?
- How did you store the passwords?
- On this project what were difficulties you encountered?
- How did you do state management?
- Did this ever have any applicable use or what was the end goal? [Usually I say to learn and to use it as a template b/c my dad runs an ecommerce business so learning payment integration and such is helpful for me]
- Etc etc etc.

And to people who are experienced btw, I know the above questions are basic questions, but to a lot of people if they've never worked in any development before a lot of people don't know the answer to those questions.

So what do you do…
You get an udemy course :) follow along and do a simple project on the side, so this way you know how to talk about it, and then you just keep studying on the side for interviews.

Go to resources… near that section will have tips on how to get cheap courses for Udemy.

Timeline

Speaking from a student perspective, if you are going for an internship, your timeline to apply is between **September 1st to September 15th. (FREAKING RECORD THIS)** AND DON'T WAIT TO APPLY B/C U NEED TO GET UR OAs ASP, and get interviewed afterwards, and acceptances are usually rolling.

For new grads, I'm really unsure, b/c I am currently in that position haha. Rising senior, I hear that it comes out late July to Mid-september, so just been checking in on those dates when it happens for the companies I am interested in.

(Update: I found out that a lot of times companies release positions early such as FB, but if you apply you actually will be relegated to the side b/c even though the position is open there is no internal openings in their system.

What this means is you should wait until late August / September 1st to September 15th depending on the company to apply.

I applied / a friend applied and immediately rejected by FB, and emailed recruiter and they said to try again late August / September 1st. I eventually reapplied under a different email, job, and new referral and while I didn't get immediately rejected like before the recruiter told me that he will follow up on September 1st, meaning that there is an internal timeline of systems and job openings that are regulated based on time.)

You should be applying when the job openings open in late August to mid September.

From there, you will get an OA (online assessment), in 2 weeks to a month, to weed out anyone who isn't prepared for interviews.

If you get past that, it is usually for interns 2 phone interviews before you get an offer.

If you are going full-time it might be that you get flown out to take interviews in person (though at the time of writing this in 2020, who knows anymore).

Some companies, depending on your school, ex. Microsoft / JP Morgan, tends to try to do as many in person interviews as they can, so if you schedule early they will send you an invite for an in-person interview at your school.

How to Study:

The optimal time to study is 3 to 4 months before September if possible, but if not, it is fine, just start studying.

I HIGHLY RECOMMEND GROKKING THE CODING INTERVIEW. If you spend money on anything this is the resource that is worth it. So my guide below will be predicated on that. (You can view the Resources section on more resources that I recommend)

Why should you study Grokking the Coding interview? B/c it breaks down interviews into 15 distinct patterns, and by "CHUNKING" the information, or dividing information like that, it gives you a systematic way to approach a problem. It isn't always going to work, but 90% to 95% of the time, you will find that these patterns will work. (if you don't like it, you do have a 2 week return period).

You need:
1 month to finish Grokking
1 month to go through Algoexpert or Leetcode (the objective here is to start covering grounds that Grokking might miss, and actually applying what you learnt by CODING IT OUT), b/c i already know that people won't be coding out everything in grokking.
1 month of mock interviews

REALIZE THAT MOCK INTERVIEWS IS VERY IMPORTANT. Being able to type out a diagram, being able to click a space bar and show a "pointer moving", these are very important minute things that you won't be used to if you don't practice mock interviews for telephone questions.

I find that if you get good at mock interviewing online, you also get quite good at mock interviewing on a whiteboard since a telephone interview needs much more organization than a whiteboard.

MOST IMPORTANT THING THOUGH…. Just start. And make it sustainable. You will go crazy, it's a lot of work to study this much, but you need to do it. So get a friend, just start 30 minutes a day if you have to, but you need to do it. It will get depressing and shit, but you got it! Watch the youtube video on the Who Am I section, lol, b/c that guy's video really kept me going.

Tips on studying / How to Study:

1) You need spaced repetition
    a) You will forget things. You lose 50% of your knowledge after the first day of reviewing, so every 4 days you must do a review on what you have studied.
    b) YOU NEED THIS, SPACED REPETITION IS HOW HUMANS LEARN
2) Study with a friend!!! You need to suffer with someone else… what can I say lol, suffering alone sucks, suffering with a friend is more bearable.
3) **Organize how you approach your problems**
    a) **https://docs.google.com/document/d/1WojFLF2HEn09sKTFoOwJv81MYxnF2pSUi4ufL7kWYYg/edit?usp=sharing**
    b) **THIS IS SUPER IMPORTANT HAVING A STRUCTURE ON HOW YOU APPROACH A PROBLEM**
    c) This is a template that I've made of my experience
        i) I try to analyze time/space, figure out what tools I have
        ii) Then I LITERALLY GO DOWN THE LIST OF QUESTIONS:
            (1) IS MY INPUT THIS
            (2) IS MY OUTPUT THIS
            (3) DOES THIS MAKE SENSE ETC
        iii) YOU NEED THIS B/C QUESTIONS AND NARROWING DOWN POSSIBILITIES ARE SUPER IMPORTANT

What to do when you get the interview:

Accept it as soon as possible if you can. If not you can technically push it back by 4 to 6 weeks for more studying, but this is why I recommend to just start studying as early as possible b/c you are giving more people chances to take spots. Also, let's be honest, if you push it back by 4 to 6 weeks, you will still be an anxious mess b/c you either understudied, and even if you have been studying, those 4 to 6 weeks might not make much of a difference unless you are just grinding those company guides on leetcode.

You should start doing mock interviews with friends, and take turns doing mock interviews. You both can throw problems at each other that you guys are familiar with, and even if the other person already knows it, it is still important to do it b/c it is more about EXPLAINING YOUR THOUGHTS and being under pressure.

What do I do When I am not passing interviews:

This could be multiple reasons. Maybe you are studying a lot or you are grinding those 1000 leetcode questions, but the problem is probably you are not consistent and/or you are not explaining your thought process well.

At the end of the day, you need a systematic way to approach problem for consistency (and also for your own sanity), but also be able to explain what is going on in your head.

Things to fix:

1) Fix your behavioral interviews.
    a) Are you sighing at any point?
    b) Are you smiling?
    c) Do you have a compelling story / way to talk?
2) Do you have a strong template to explain your thought process when you go through the interview?
    a) **https://docs.google.com/document/d/1WojFLF2HEn09sKTFoOwJv81MYxnF2pSUi4ufL7kWYYg/edit?usp=sharing**

Tips on How to Get Cheap Udemy Courses

1) Udemy.com is a great place to get courses
2) They are on “sale” all the time between 9.99 to 12.99. They have dynamic pricing models so if you are a new user “suddenly a sale” or that sometimes they will have “surprise sale” from 199 to 9.99 lol. If you ever see it not on sale just go on incognito.
3) If you have a VPN you can set yourself to turkey -> will convert the cost to 24.99 turkish dollars for a **new account**. After you make your first purchase will lock the location. If you open up VPN on your phone using the Udemy app, a new account too, will also automatically show in USD pricing instead of turkish pricing, 3.99.
4) You can GIFT your main account the course too lol instead of purchasing it to your account.
5) Also Udemy has the most generous easiest return policy of 30 days. You just copy and paste the link for your course and they will refund you. I tend to keep the courses as references though for the future and b/c i appreciate the instructors, lol.

Resources:

HIGHLIGHTED RESOURCES ARE #1 RECOMMENDATIONS!!!

Side note: Resources are definitely worth the money and will be paid back a lot if you end up getting a job anyways :) Don’t be stingy, you are a software engineer, and making probably the top salaries in the US even if you don’t get into FAANG.

This is an academy my one of the Udemy instructors that I love alot. If you are thinking about doing multiple courses I recommend this person:
https://bit.ly/2QdRWfM

Great way to do mock interviews online:
https://www.pramp.com/dashboard#/schedule

Facebook Groups:
Subtle Asian Tech + Subtle Asian Networking

Under the mentor tab there are a lot of people you can message to ask them questions, and people are kind enough to usually give you advice on how to prepare, what they do etc. You don't need a referral, what you need is information. Also being a part of a community is very important! And being able to ask questions is quite good too :)

Grokking the Coding Interview on educative.io
Note: Grokking on educative.io has now turned into a subscription based service versus paying 1 time. B/c of that I recommend you to ask friends to split the cost, or to balance what you think is worth the cost or not. They do have a refund policy, you can look into so look into that if you end up not finding it is worth it.
One of the best systematic ways to approach coding interviews. It really prepares you in patterns on how to approach problems. Some of the ways are not optimal, but it does give you a tool to fall back on. And you can use it in collaboration with youtube videos or other sources in order to maybe see other ways to do it.

Grokking the Dynamic Programming Interview
Probably one of the best resources in terms of understanding Dynamic Programming. It definitely takes a while for it to click, and it took me a bit to understand what is Knapsack and what is the pattern it was trying to explain, but once it does, things begin to click. But tbh, dynamic programming questions at this difficulty is quite rare, and almost never comes up.

Algoexpert.io
While it isn't structured as much as Grokking the coding interview, it has an excellent question bank, explanation, and very helpful crash course to data structures / algorithms for those who are not well acquainted with O(n) analysis. However, I would suggest someone to do more analysis study on the side if you can't do it. But definitely a great resources to use in conjunction with Grokking the Coding Interview.

Leetcode…
This is not my favorite resource. But once you are ready, you can go through the company guide for premium and those are quite helpful. The problem with leetcode is there is no solid explanations even through forum discussions, and just doing a lot of leetcode doesn't equate to consistency.

50 page Guide to Getting Gud:
This is an amazing guide written by someone named Edbert from a FB group, he is an amazing person and wrote a lot of details on studying here
https://docs.google.com/document/d/1eKirumpmwDWTtKCJKn2HuoQ2NavEfR41whmTyaQcio4/edit?usp=sharing

Linkedin Recruiter Networking Sheet (Sheet with 2K Technical Recruiters from different companies):
https://docs.google.com/spreadsheets/d/1Lhs-F-KEzyw0XmJd0P23-KiMh1RJ6Zj2SN5EDuOQ3XM/edit?usp=drivesdk

Udemy course on Data Structures / Algorithms
If you have never done Data Structures / Algorithms before, I recommend that you take Andrei Neagoie (Javascript) or a Python Data Structures / Algorithms course.

I personally think that Python is the best way to go, but I feel that any course you have, if you feel comfortable with programming it is very easy to switch between languages like Python, so even if a course is in Javascript you can translate the code. But if you do want a specific course with Python, can search on udemy for such a course.

Just get any course 4.5 stars or better.

Looking at those 150K to 300K salaries as Motivation! LOL
https://www.levels.fyi/

Q/A

What is my opinion of CTCI / EPI, and why do I recommend doing Grokking / Algoexpert / Leetcode.

Ok, the first thing is that Grokking gives a systematic breakdown. Compared to all these other resources, it is the only one that I know of that breaks things into distinctive patterns which results in consistency and ability to understand approximately where things begin to fall intuitively.

The next thing is that CTCI is very limited in scope of it's languages, is older, and while still a great book, Algoexpert has video explanations and support for many different languages and built-in IDE. Basically, they both accomplish the same objective, but Algoexpert is just a better version of CTCI on every aspect.

The final thing is that EPI, or elements of programming interviews, is extremely dense. While it definitely is a great book, I wouldn't recommend this to beginners b/c it isn't friendly and that Grokking + Algoexpert will basically cover things to a similar degree. I would say that EPI is more of it you need a challenge beyond Grokking / Algoexpert if you ever reach that level.

Thus:
Grokking for systematic approaches and breakdown
Algoexpert for question banks and great explanations of problems that are very common and popular
Leetcode Premium for Company Guides / Questions pertaining to those companies

What about System Design?
https://www.educative.io/track/scalability-system-design-for-developers

I recommend the above track along with Algoexpert System Design course.

If you are a student, you can search up, Educative.io student pack and it is included if you have a Github student pack for everything that is a non-interview course. So I would recommend to read the Web architecture 101 section of the system design track for free, and then go to Algoexpert and take the System design course there, and take the System design course on Educative.io