THE IEEE CS INTERVIEW GUIDE

Getting to the Interview

- **What's a good number of companies to apply to? What if it's my first internship?**
    - For your first internship, 50 is a good minimum number of companies you should apply to. While you should not just be applying randomly, applying to several companies can increase your chances of getting your first internship. Once you have one internship, you can be more focused, but 30 is still a fairly safe number of companies to apply to.
- **Where should I look for jobs?**
    - LinkedIn
    - Indeed
    - Glassdoor
    - Handshake
    - ECAC
    - Piazza
    - Google Jobs
    - Job fairs
- **How do I determine if I am a good fit for a job?**
    - Always read the job description of every job you apply to. There are two things to consider when determining if you are a good fit for a job:
        - Will I enjoy this job: the job description will generally give you an idea of what you will be doing.
        - Do I have the necessary skills for the job: read through the skills the job requires. Based on the job description, try to determine which ones are the most important ones to have. If you have most of these important skills on your resume (at least 50%), the job is a good fit for you.

- **This internship says it wants 5 years of work experience and 15 years of experience with Kubernetes! Even if I have all the required other skills, should I apply to this job?**
    - Something very important to remember about job descriptions is that they are like *wish lists*. A hiring manager writes skills they desire from a prospective candidate. Just because you don't have 100% of what a job description wants does not mean you should not apply.
- **How should I structure my resume? I have a ton of different achievements, so should I just dump all of them onto my resume?**
    - No. You want your resume to be structured to show your aptitude for the job you are applying to. Having a variety of achievements and skills is great, but you want them to show a "spike" and not several small peaks. A resume shows a "spike" if a recruiter can look at it and easily identify a strong interest or passion. If you have several different achievements, know when to put each one on your resume and when to leave them off.
        - Here is an example. Suppose Jim and Bob are both applying for a software internship position. Jim puts 3 good software projects on his resume. Bob puts his internship doing verification, his research in a nanotech lab, and a software project on his resume. In this case, *Jim would usually have the stronger resume because the image his resume creates for a recruiter is someone committed to software engineering*. Meanwhile, while Bob might have multiple talents, recruiters will not be able to pick up on a passion or specialization he has when reading his resume.
    - ECAC is a resource you can use to help you to structure your resume. They also have resume reviews if you need to get advice about your current resume.
- **What are referrals? How can they help me land an interview?**
    - While the specifics of a referral vary between companies, a referral is generally when an employee at a company recommends you in some way.

Sometimes, they will recommend you for specific positions, and other times they will actually complete a job application for you.
    - Referrals are an excellent way to get ahead in the hiring process. Getting a referral for a position greatly increases your chances of a recruiter looking at your resume. This is important as one of the main reasons you might not hear back from a company you applied to is because your resume was not looked at.
    - You should try to get referrals for the jobs you are most interested in.
    - A great way to get referrals is to connect with upperclassmen and your TAs on LinkedIn and network with them while they are UT students. Then you can ask them for a referral after they graduate.
- **What is LinkedIn? What are some uses for LinkedIn?**
    - LinkedIn is a professional networking/social media platform that allows you to connect with companies, alumni, and current students and document your professional experience.
    - Recruiters and people who refer you will sometimes look through your LinkedIn to see your other skills. Additionally, it can be used to connect to and talk to alumni and recruiters!
    - It is good practice to update your Linkedin profile whenever you have finished a project, got an offer, or when you have gotten a new leadership position.
- **Why are projects important on a resume, particularly for students looking for their first internship?**
    - When someone is looking for their first internship, most of the time the experience section of their resume is empty. Projects basically allow you to showcase your skills and also give you the “experience” you need for recruiters to notice your resume.
    - Projects *do not* have to reinvent the wheel and *do not* have to be building something from scratch. Something as simple as a task scheduler project can be enough to get you an internship. Both strong class projects and strong personal projects can help you stand out.

    - If you have multiple projects, you should decide which projects are more relevant to the skills that are on the job description and put those projects on your resume.
- **How can I be successful at EXPO? What's a good way to network and build connections with recruiters/engineers?**
    - EXPO is a job fair held each semester where students can network with employers and possibly get jobs/internships. The event is hosted by ECAC.
    - Before EXPO:
        - Study companies that are coming to EXPO that interest you. Some easy ways to study companies are:
            - Check their Wikipedia page and company website
            - See what their products are
            - Look at jobs they have posted on job boards and see if you qualify.
    - During EXPO: Try to connect with as many engineers and recruiters as you can. Make sure to get business cards or contact information from every person you speak with so you can reach out to them later.
    - After EXPO: Contact everyone you met during the fair and thank them for their time. Follow up with any questions you might have come up with since your meeting with them.
- **What are some other good ways to network?**
    - ECAC and Handshake events are a great way to connect with recruiters and engineers from different companies.
- **What's the difference between a hiring manager, a recruiter, and an engineer?**
    - Hiring Manager: The person who generally decides who gets hired for a job posting.
    - Recruiter: Usually an HR representative and in charge of recruiting students.

    - Engineer: Networking with engineers at a company can be helpful for getting referrals. However, engineers are generally limited in how much they can influence the hiring process.
- **What's a good way to reach out to recruiters? Should I use their LinkedIn or their email? What if I can't find their email?**
    - If you have a recruiter's email address, it is always preferable to use that instead of their LinkedIn.
    - You can get more information about contacting recruiters here: https://www.cscareers.dev/blog/landing-software-engineer-interviews-through-cold-emailing
- **Can I get ahead in the recruiting process through cold emails or cold messages on LinkedIn?**
    - Yes! A key to success in the job search is to not be afraid to reach out to people who aren't familiar.
- **Why is using a uniquely designed resume a bad idea?**
    - An ATS is a software system that, simply put, parses a resume and matches its similarity to a job description. In today's job market, 75% of resumes are never read by a human. Therefore, it is important to create a resume that the ATS can easily parse.
    - Uniquely styled resumes might not be ATS-friendly, and by using them you could miss out on an interview for a job you are qualified for.
- **If I get an interview, how can I find out what kind of interview it is (behavioral only, behavioral + technical, etc)?**
    - You should email the person that contacted you about the interview. If you don't get a response, then you can go to Glassdoor and search up the company. There are usually reviews about the interview process that you can use to determine what kind of interview it might be. You can also just study for both kinds of interviews if you are still unsure.
- **What are some good resources?**
    - McCombs resources

- https://www.cscareers.dev/blog/landing-software-engineer-interviews-through-cold-emailing
- ECAC (or Handshake for CS students)

Phone Screen or Behavioral Interview

- **What are good ways to research a company prior to an interview?**
  - The best way to research a company is to go to EXPO/networking events that the company is going to and talk with recruiters and engineers about the company. Additionally, connect with alumni that work there and ask to have a coffee chat with them to learn more about the company
  - If you can't do this then first try to start with the company website and Wikipedia. Also check the news regarding the company to be updated on recent acquisitions and product releases.
- **What are good strategies for behavioral interviews?**
  - Throughout the semesters, you should start compiling a list of stories you can use to answer behavioral interview questions. If you compile them now then later you can still share the story in greater detail than if you tried to remember it later.
    - Examples: working on team projects, fixing bugs in code, studying for classes.
  - Before the interview, try to create a list of questions and answers for each interview. Make sure you use the STAAR method to construct your answers and make sure your answers are maximum 2 minutes long.
  - If you have time, ask a friend to interview and give you pointers on how to improve your answers.
- **What is the STARR method? How can I use it to construct behavioral interview answers?**
  - The STARR method is a format you can use to construct answers. STARR is an acronym for Situation, Task, Action, Result, Reflection. Situation provides background and context to the situation. Task describes the problem and why it is important. Action refers to what you did, how you did

it, and what skills or tools you used. Result/Reflection explains the results and outcomes of the action and what you learned from it. Try to include a soft skill such as teamwork or communication in your Result/Reflection.

- **What are some good resources to learn about or study for behavioral interviews?**
    - McCombs resources.
    - One of my favorite articles on interviews. The “Behavioral Questions” section is particularly insightful.
    - You can search up the company Glassdoor to see if people have posted what questions were asked.
    - If you have a hard time finding behavioral questions, try to create answers that illustrate a skill instead of an answer that answers a specific question. This can help when you get a question you have never heard before. Some skills to illustrate are teamwork, leadership, problem-solving etc.
- **The interviewer asked me a really specific question about a scenario that has never happened to me. What should I do? (EX: “Tell me about a time when you were in a meeting and had several people saying multiple different things at once and you had to make a critical decision in a short amount of time”).**
    - If this happens, be honest with the interviewer and say that the specific situation has not happened to you. Then talk about what decisions you would make and how you would respond *if you were put in that scenario*.
    - Alternatively, you can talk about a tangentially related scenario and what you did there.
- **How technical should my answers be in a behavioral interview? Does this depend on the interviewer?**
    - Try to follow the STAAR format and include technical aspects only if it is important/relevant to the answer. Behavioral interview questions are used to gauge your soft skills, not your technical skills.
    - Additionally, if the interviewer asks for more information then you can elaborate on the technical aspects.

- **If my interview is on HireVue or some other recorded interview platform, how should this change my approach to the interview?**
    - Try to look at the camera during your interview
    - Right before each question, put on a smile.
    - If you are given multiple choices to answer, it's ok to use the time of the first answer to contemplate further if you're unsure what you want to say to a question.

Coding Challenges and Technical Interviews

Note that "Leetcode" here can be replaced with Hackerrank, AlgoExpert, etc...

- **What is a coding challenge?**
    - Companies send out coding challenges to weed out applicants before doing actual interviews. Completing these challenges is vital to advancing further in the interview process.
- **What is Leetcode? How can Leetcode help me prepare for a technical interview or coding challenge?**
    - Leetcode is a coding platform that allows you to practice coding problems that companies ask during technical interviews and coding challenges.
    - It is a useful resource because it has a lot of problems and solutions that you can practice.
- **How can I determine what coding questions that a company might ask?**
    - It is hard to determine which questions you will get asked. However, Leetcode Premium allows you to filter problems that companies have asked before.
    - If you do not want to get Leetcode Premium, check out cscareers.dev on Discord. You can find a company's most common coding questions by using the leetcode-bot channel.
    - You can find questions a company has previously asked on Glassdoor.
- **How can I improve my ability to code while speaking to an interviewer?**
    - Practice doing technical interviews with a friend.

    - Additionally, you can try to think out loud when practicing problems online. During a technical interview, you will have to communicate to the interviewer your thought process and thinking out loud can help you code while speaking to the interviewer about your solution.
- **What should I do if I am unable to find the optimal solution during a technical interview?**
    - You don't want to run out of time for the interview. If there are more questions that the interviewer wants to ask you, move on with the solution you have.
- **Aside from Leetcode, what are some other resources that can help me practice coding problems?**
    - Hackerrank
    - AlgoExpert
- **Do I need to use Leetcode to get any software job?**
    - No. There are plenty of companies that do not use Leetcode-style problems when hiring engineers.
- **What coding language should I use when practicing coding problems? How about during a technical interview?**
    - You should use a language that you are comfortable coding in. Some coding challenges sometimes ask you to code in specific languages but most companies will let you decide which language to code in. The language you use should preferably be well-known (EX: Don't use ARM assembly language).
        - Also, it is good to pick a language that you know the intricacies of. For example, for many embedded systems interviews, it may be beneficial to choose a language like C or C++ where you can show extensive knowledge over pointers and memory management.
        - On the other hand, if you have a lot of experience using Ruby on Rails for web development, then you should use Ruby.
    - If you don't have a specific language you prefer, Java, Python or C++ are generally good.

Leetcode Preparation for Technical Interviews

Note that "Leetcode" here can be replaced with Hackerrank, AlgoExpert, etc...

- **Should I get Leetcode premium? (NOTE: IEEE CS at UT is not affiliated with Leetcode). What are a company's "frequently asked questions"?**
    - Frequently asked questions are questions that the company has asked during its coding interview. Leetcode premium, you can see a company's frequently asked questions.
- **What is a good method for practicing problems and improving my skills with data structures/algorithms?**
    - First spend some time reviewing theory (big-O complexity of common algorithms & data structure operations, types of data structures & usage pros/cons, etc.). Then, do some more basic practice problems from online sources to identify which types of topics and problem structures you need to work on the most. For each of the more difficult topics/problems, spend some more time devising unique and resourceful potential solutions—even if incomplete or incorrect, the process of diving into a difficult problem and figuring out the intricacies required for a satisfactory solution can be more educational than doing tons of basic problems. Finally, over multiple days, have a peer quiz you using small randomized sets of questions, simulating real interview timing as much as possible.
- **After how long should I give up on a Leetcode problem and look at the solution?**
    - There is no universal time limit—in general you can pace yourself as an interviewer would, by stopping after 15 or 30 minutes of work based on the difficulty of the problem. If you have more time to prepare, (ie. if you are drilling problems in a relaxed environment instead of simulating an interview environment with fewer questions), then it is valuable to attempt approaching the problem with a variety of solutions, as this process ensures you will remember how to arrive at a solution with limited information. However, once you have picked up something of value from

the problem—such as the core algorithm, or what pitfalls to look for when answering similar questions—it is worth moving on quickly.

- **I feel like I'm not improving fast enough or not improving at all. What are some common pitfalls people fall into when practicing coding problems?**
    - One common pitfall is relying on the answer for the solution. It is easy to become dependent on the ability to check your correctness while you code, which is why it is essential to attempt multiple approaches before giving up and looking, else you might feel stumped in the actual interview. Another common pitfall is getting caught up on the details of one particular sub-problem. Often when devising an algorithm or verifying a potential solution, we can spend a lot of time trying to prove that something could possibly work; it is important in these situations to step back and evaluate whether the task at hand is worth spending such time and effort, or if it would be more valuable to put a pin in the current work and look at the problem another way.
- **I have an interview at a company and I know the top questions the company asks. How should I practice these questions? Should I try to come up with variations of problems?**
    - Practice these questions alone, for your own education and research, but also with peers, as collaboration can benefit everyone when attempting the more difficult questions. As for variations, that is an excellent idea, and peers can help with the variations and their evaluations as well. Often interviewers will ask you to create a solution to a problem, then to optimize the code you write to make the solution more efficient; this is an example of a great way to add variety and complexity to existing practice problems.
- **I got an interview with Amazon (or Google, Facebook, Uber, etc) and they have so many frequently asked questions. I don't have time to go through all of these. What should I do?**
    - You can't do them all, and even if you did there's a chance the question they give you in the actual interview will be completely unrelated. However, since there is also a chance it will be related, it is worth picking a

few familiar and a few unfamiliar questions from such a bank, and using them as practice instead of some other question bank from online. Often these frequently asked questions will be a more difficult sample, so you shouldn't spend all your time on these questions, but you are better off having practiced a few of them to round out your repertoire of problem solving techniques.

- **How many of a company's frequently asked questions should I do before an interview?**
    - As many as you feel comfortable, just not all of them. Focus on the ones that are specific to your weaknesses and unfamiliarities, and also try to do a few from other sources. More importantly, spread the questions out over the course of a few days.
- **Should I worry about hard questions, or just medium and easy questions?**
    - As for a spread of difficulty, it is worth keeping it evenly distributed to keep your mind active and refreshed. But rather than the objective level of difficulty, focus on topics and types of problems that have been difficult for you in the past.
- **What are some good Leetcode resources?**
    - 14 Patterns to Ace any Coding Interview Question
    - Blind 75 questions
    - cscareers.dev