* How to get the Interview
  + Besides following resume tips, make sure to apply to as many places as you can. To get our first internships, we recall applying to approximately 250-300 places before we secured our internship plans for that summer. Also, if you do get ghosted, don’t take it personally, usually, university recruiters often spend so much time reviewing a lot of applications.
  + Other precautions to take to get noticed are to try attending career fairs if you can, you might be able to get an interview(worst case, free swag!). Other than that, try reaching out to upperclassmen or friends you know that interned and ask for referrals. It’s one of the best ways to get noticed!
* What to expect
  + Behavioral Interview: These interviews ask you questions about culture fit such as “Why are you a good candidate” and “Tell me about a time when you ...”
  + Coding/Technical Interview: These interviews ask you questions similar to what you see on Leetcode and Hackerrank. These interviews are designed to test your Data Structures and Algorithms knowledge.
  + System Design: System Design involves the interviewer testing your building to design a service/software and test your knowledge of understanding what things to use for the task and how you will integrate them together. You’re definitely not expected to know this and it’s not likely you’re gonna get asked this. Places that could ask you this are Unicorns, Trading Companies and Hedge Funds, and Big Established Companies.
  + Concurrency/Low Level Interviews: If the company’s biggest product involves low level principles such as networking principles and kernel stuff, there’s a possibility you can get asked this. Places that come to mind are hardware companies and trading firms.
  + Some companies may adopt only one of these interviews and some may adopt all.
* How to Ace the Interview
  + It’s highly recommended that you look up the interview experiences that other students have faced so that you can potentially filter out companies with red flags and know what questions to expect. Common sources to search up on this would be Reddit ([r/csMajors](https://www.reddit.com/r/csMajors/) and [r/cscareerquestions](https://www.reddit.com/r/cscareerquestions/)), Jumpstart (Relatively new portal for students), [Glassdoor](https://www.glassdoor.com/index.htm) and maybe [Blind](https://www.teamblind.com) (Aside from the toxic TC or GTFO culture, they do give good advice on interviews). With that said, here’s some advice we have when you approach each kind of interview we’ve seen.
* Advice on Behavioral Interviews
  + Use the [STAR method](https://www.thebalancecareers.com/what-is-the-star-interview-response-technique-2061629) when describing your experiences. Being quantifiable with the impact of your actions will impress the interviewer.
* Advice on Technical Interviews
  + Begin by reviewing your notes from the Data Structures and Algorithms class. Do not proceed further until you know how to implement these DS and As from scratch with the language of your choice(If you do know python, it’s recommended as there’s a lot of builtin features!). After doing so, we highly recommend a book like CTCI and EPI to gain a review on programming language details and your DS and As. Then, visit sites like LeetCode to practice real questions from major companies. A Facebook Engineer completed 600 LC problems and compiled the most important ones into a [list](https://www.teamblind.com/post/New-Year-Gift---Curated-List-of-Top-75-LeetCode-Questions-to-Save-Your-Time-OaM1orEU) here. During the interview, make sure to talk out loud about possible approaches and tradeoffs before whiteboarding. It is perfectly acceptable (often recommended) to ask the interviewer to ask questions about the problem and get clarification. Once you have an idea in mind and have talked about it with your interviewer, begin whiteboarding. While you talk about the final idea you want to use, write out pseudocode and comments about all the steps you need to implement in order to finish coding your solution. After that, start coding. Make sure to have proper function headers, syntax, spacing, classes/structs, imports, etc. After coding your solution, give a brief explanation and attempt to make it run with less space and in less time (if your solution is not as efficient as you think it can be).
* Advice on System Design
  + These are somewhat hard to approach if you don’t have experience ever doing it. If you do have experience designing and building services in your spare time and as part of your work experience, definitely rely on your experience. An important thing is to definitely ask clarifying questions. There might be hidden requirements you didn’t think about that could drastically change the way you approach the solution.
* Advice on Concurrency/Low level
  + Understand basic principles such as Processes vs Threads (A lot of people don’t know the difference!) TCP vs UDP and how to make an application thread-safe. Other than that, it’s recommended that you familiarize yourself with basic OS concepts such as Deadlocks, locks that you can utilize to make an application thread-safe, etc.
* Searching for Full-Time Jobs:
  + The big bucks.
  + The process for finding a Full-Time Job is usually very similar to finding an Internship. There’s three main differences are:
    - Harder Questions. Ex: Google usually asks Leetcode Mediums to Hards + the special Leetcode Hard question that Google asks it’s applicants (they create a new one every year).
    - More Rounds of Interviewing: For example, Microsoft makes interns do 2 rounds while New Grads do 4 rounds during the onsite part of the process.
    - Compensation: Interns usually get an hourly rate and, possibly, a housing stipend. New grads, however, are given a yearly salary and, possibly, a sign-on bonus, stocks, and benefits e.g. health insurance, vacation days, etc.
  + The process for finding a Full-time Job won’t really change as much as finding an internship, but keep in mind that the bar is higher. This is probably the biggest reason why you should look into interning early; by getting an offer at the place you like, you don’t need to go through the daunting process of finding a full-time role.
  + Get as many offers as you can this time around, so you can negotiate and select the position, company, compensation, and location that works best for you.
* Negotiation
  + Negotiation is a really powerful tool that you can use in the interview process, even as an intern. There’s a lot of guides to negotiation and we recommend Nick Singh’s guide (Look at his LinkedIn and newsletters) for more.