CSE 373 SP21 Section 9

Interview Prep

Interviewer: What's your name?
Me: Hired.
Interviewer: You're Hired?
Me: Thank you, sir.
You are already under my genjutsu

Special thanks to Kimberly Nguyen and CSE 492J, whose slides we've copied and butchered completely.

MicroTeach: Introduction to Recruiting and When to Apply

The Recruiting Process (specifically SWE* Positions)

- The most important time: **Fall**!
  - August-November
- Internships and jobs will be recruited for until positions are filled
- Come winter (January-March) there will still be some jobs available but a smaller amount
- **Use the time you have now to (the rest of spring and summer) to start building relationships with recruiters and practicing coding!

*SWE = software engineering

A month by month plan for recruitment season

- **<u>August</u>**
  - Make a list of companies you're interested in
  - Apply to companies online!
    - Online job boards
    - Linkedin (make a profile!)
    - Google positions! (ex: "software engineering internship summer 2021")
  - Practice coding questions
    - Leetcode, Cracking the Coding Interview, etc.
- **<u>September</u>**
  - Network/talk to friends about references
  - Email recruiters
    - Look up "company_name recruiter" on Linkedin
  - Continue applying!
  - Continue practicing coding questions!

A month by month plan for recruitment season

- **<u>October</u>**
  - Attend career fairs
    - Allen School Career Fair, Diversity Career Fair, iSchool Career Fair, UW-Seattle Winter & Spring Job & Internship Fair
  - Attend company events
  - Follow up with recruiters
  - Begin the process of interviewing
- **<u>November</u>**
  - INTERVIEW!

A big NOTE about this timeline!

DO NOT lose hope if you don’t get an offer in Fall.

Many companies will still be recruiting in Winter (and even Spring) quarter. Work those connections (including the TA talking to you **right now**), keep applying to job boards, and hang in there!

The requirements to getting the interview!

- **Build a resume**
  - A one page reflection of how you use your time outside of class
  - Highlight skills and experiences
- **Apply to companies' online listings**
  - Check job boards, Linkedin, Google
- **Meet with companies on campus**
  - Attend career fairs or company talks
- **Network!**
  - Ask for referrals
    - Example: the TA speaking to you.
- Follow up with contacts & recruiters

Crafting Your Resume- Key Points

- Make it **one** page
- Make sure there are **NO** typos
- Required sections:
  - Contact
  - Education
- Additional (or optional) sections:
  - Objective
  - Work experience
  - Programming experience
  - Leadership experience
  - Skills

The Realities of being a Non-CS/CSE Major applying for CS Roles

The recruitment process is challenging for everyone but will be challenging in a very different way for a non-CS/CSE major applying to these roles! Be prepared for rejection but also just be prepared, period.

The Realities of being a Non-CS/CSE Major applying for CS Roles

- **Make it clear that you are interested in a software engineering position**
  - Add an objective to the top of your resume
- **Explore classes outside of CSE373 to highlight further CS class experience**
  - CSE374, CSE154, CSE163, Coursera courses, iSchool courses, etc.
- **Acquire significant outside experience in coding**
  - Do a personal project (build a website, app, etc.), contribute to open-source projects, join RSOs that provide coding experience (Impact++, etc.)
- **Acquire experiences that give formal credibility to your coding ability**
  - TA for a CS course (CSE373 :D, CSE14X etc.), get a software engineering internship, etc.

But how do I get “experience”?

- **How do I get coding experience outside class?**
    - **Join an RSO now** that involves coding- this actually looks *really* good.
    - Another good idea: join an RSO and literally offer to build them a website. (Joyce can verify that this worked for them).
    - https://www.firsttimersonly.com
    - Do this **now** so that in the Fall you are ready and have **cool projects** to talk about!!!

Warnings for International Students

- There are some **stupid, silly** rules you need to watch out for.
- May need to have a **major declared** to intern in order to get approval from educational advisor as part of your visa requirements.
- Many companies will also **refuse** to hire international students because of sponsorship requirements- check the listing before applying so you don’t waste your time.
- **Definitely** still doable to get an internship/FTE role! Many international 373 TAs have done this (in fact *all of them* have).

Resources

**The information provided in this MT is courtesy of Kimberly Nguyen

- **CSE492 J** (if you're able to get in)
  - Career prep seminar in Fall
- **Kim's Resume Guide:** http://bit.ly/cseresumeguide
- **Kim's Career Hunting Guide:** https://docs.google.com/document/d/1chTrHwSC2iv_EU-Br55Hh4YhdozQ7R2RaWVHusMZUul/edit
- **373 Crowd-Sourced Resource List** https://docs.google.com/document/d/1GCyUKCUnIW_doc-LjDuYEm4D9mk3K6hiHnATRoJzOcA/edit?usp=sharing
  Note: Includes cool RSO to join and gain experience with, dank interview questions we found, etc.

Microteach: The Behavioral Interview

When will you be asked behavioral questions?

- **Recruiter Screens**

  In first round screens with recruiters, questions are usually exclusively behavioral.

- **During Technical Interviews**

  Technical interviews with engineers are usually kicked off by 1 – 2 behavioral questions.

Why it matters

- **Learn about your previous experiences.**

  Interviewers want to know how you behaved in a real-world situation.

- **Learn about how you've made an impact.**

  Interviewers want to understand the measurable value you added to that situation.

- **Learn if you are a good fit for their company, role and team.**

  Interviewers are trying to learn how you define something like, "pressure at work" and if that aligns with their team's perspective.

Common themes for questions

- Teamwork
  - Give me an example of a time you faced a conflict while working on a team. How did you handle that?
- Ability to Adapt
  - Tell me about a time you were under a lot of pressure. What was going on, and how did you get through it?
- Time Management Skills
  - Tell me about a time you had to be very strategic in order to meet all your top priorities.
- Communication Skills
  - Give me an example of a time when you were able to successfully persuade someone to see things your way at work/school.
- Motivation and Values
  - Tell me about your proudest professional accomplishment.

Prepare

- Read the job description
- Read the company’s mission statement/value statement
- Email the recruiter 2 – 3 days beforehand and ask them if there’s any insight they can give you into the team you are interviewing with or if they have any last minute tips for you.
- Hit up your friends/acquaintances/network that have interviewed at or currently work for the company that you are interviewing for and ask for tips.
- Scour the internet for intel (reddit, glassdoor, fire up the search engine, etc)

Articulate: The STAR Method

- **Situation**: **What is the context of your story?** In setting the situation, you are telling your listener when or where this event took place. For example, “We were working on a quarter long project in our upper division Machine Learning course…”

- **Task**: **What was your role in this situation?** For example, “It was my role to lead the effort to determine which machine learning model would be best to use”

- **Action**: **What did you do?** For example, “I researched several machine learning models and narrowed it down to 3 that might work. I worked with my teammate who oversaw setting up our initial data set to run a few comparisons. I put together a comparison report with metrics to make it easy for the team to discuss and collaborate on next steps.”

- **Result**: **What did your actions lead to?** For example, “We were able to choose XXX algorithm for the project which lead YY% accuracy. It was incredibly rewarding to navigate a new technical problem space and work together with my teammates.”

Pro Tips

- **Honesty is the best policy**

  Be as open and honest as possible with your answers. As you review possible stories, remove any where you can’t quite remember all the key details or those that make your uncomfortable to tell.

- **Practice makes perfect**

  You need to practice your interview responses aloud. In the mirror to yourself or better yet, with a friend so you can get feedback.

- **Two minutes**

  Keep your initial answers to under two minutes. Don’t worry, if don’t cover everything your interviewer wanted to know, they will probably ask a follow up question about it.

2 most important questions

- **Tell me about yourself**

  Who are you?

  What are your interests? Goals?

  Why are you interested in this position?

  This should be 30 – 60 seconds

- **Tell me about a technical project**

  At any given time, you should have at least 1 (if not 2) well crafted pitches ready to go for your most significant coding projects. Pick your biggest or most technically interesting project(s). It’s okay to talk about school projects.

Project Discussion Example

"I spent this summer working at an advertising network, specifically trying to drive engagement on our video ads by A/B testing new ad content and formats. I worked primarily in the backend and used Python and R for data analysis. I produced an 8% improvement in click-through rates across the board over six weeks of testing."

Follow up questions:

- How long did you work on this project?
- How big was the team working on this, what was your role specifically?
- Why did you choose that technology stack?
- What was the biggest bug you encountered and how did you fix it?
- If you redid the project what would you do differently?

End with your own questions for the interviewer

- Having a good question shows you are interested in the position.
  - What is your favorite part about working for X?
  - What keeps you excited and motivated to show up to work every day?
- This is your opportunity to probe the company and decide if it’s a good fit for you.
  - Reflect on your values and come up with some non-aggressive questions you can ask.
  - Large tech companies can sometimes have a reputation for being ultra competitive. Do you have any perspective or advice for a young, hungry new grad that cares about work but also cares about life outside work?
  - Covid-19 has brought to light how important it is to work for an employer that takes are of its people. How did X make you feel safe and valued during the situation?
  - Growth has always been something that is very important to me. How has company X supported your growth both personally and technically?
- Don’t ask rude questions
  - Did I pass the interview?
  - How much do you make?

MicroTeach: The Technical Interview

Why technical interviews suck

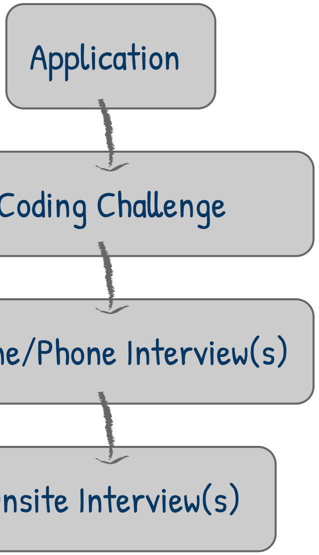
- They are **hella stressful**
- We don't even know if they actually **work**
- Your interviewers are probably not THAT qualified
- They are almost always super awkward

Why technical interviews suck

- They are **hella stressful**
- We don’t even know if they actually **work**
- Your interviewers are probably not THAT qualified
- They are almost always super awkward

But every company does them….so you gotta play the game.

The Typical Interview Process



Typical Technical Interview Structure

1. Introductions (Under 5 minutes)
2. Project discussion (5 minutes ish)
3. **Coding question (45 minutes ish)**
4. Your questions (5 minutes ish)

1. Introductions

- > “Hello.”
- > “How are you?”
- > “Who are you?”
- > “Why do you want to work here?”

Be nice, be polite, be confident.

Firm handshakes never hurt!

(Or elbow bumps?)

2. Project Discussion

Have 1 or 2 project(s) that you feel confident talking about in front of people who know what's up.

> "Are school/internship projects okay?"
>
> ○ Yes, but prefer things you did outside of **strictly** course work.
>
> Don't be afraid to brag about what you and your team did, but be prepared for questions.
>
> "We" is a good word, but don't overuse it.

3. The Actual Coding Part

Be patient, we’ll get here ;)

4. Your Questions

Yeah I was also surprised that this part **actually matters**.

> Show your interest

> Learn more about the company

Prepare some questions before you go in, and make sure to ACTUALLY listen to their answers.

Tip: If you don’t ask **any** questions, you look **really bad.** Have at least some prepared before you go in.

MicroTeach: CEAOW-IT!

CEAOW IT

1. Clarify
2. Example
3. Approach
4. Optimize
5. Walkthrough
6. Implement
7. Test

Practice question!

Given a number n, print n-th Fibonacci Number.

1. Clarify

> Clarifying questions

a. Can n be a non-positive number?
   i. Depends, n can be 0, but not negative.

b. Can we use additional data structures?
   i. Yes, assume we want the fastest overall runtime.

c. What should be the result when n = 0?
   i. The result should be 0, before the first 1 in the sequence 1, 1, 2, …,Fib(n)

2. Example

Once your interviewer has clarified this super vague question, you should be able to produce a few **example** inputs and outputs.

Edge Case 1:

n = 0; Fib = None

Output = 0

Edge Case 2:

n = 1; Fib = 1

Output = 1

Middle Case 1:

n = 2; Fib = 1, 1

Output = 1

Middle Case 2:

n = 9; Fib = 1, 1, 2, 3, 5, 8, 13, 21, 34

Output = 34

Keep these examples handy! We'll also use them when we get to **T (Test)**!

3. Approach: Brute Force

SLOW DOWN. Again, SLOW DOWN.

Before you start writing code, try to think of **any** solution to the problem, *even* one that's super slow and terrible.

This is good because even if you can only come up with the crappy, slow solution to the problem, you at least have **something** written down.

If you're **running out of time**, make sure you always code down *something,* even if it's not optimal.

(Your TAs have passed interviews by remembering this).

3. Approach: Brute Force

Problem: Given a number n, print n-th Fibonacci Number.

Example Brute Force Solution:

1. Use a recursive function to solve
2. Starting with n and descending down, recursively return the addition of the last and second last numbers of our sequence
3. End our recursion when we hit our base case n = 1
4. Has O(2^n) runtime with O(n) space complexity…

3. Approach: Brute Force

Problem: Given a number n, print n-th Fibonacci Number.

```
public static int fib(int n) {
    if (n <= 1) {
        return n;
    }
    return fib(n-1) + fib(n-2);
}
```

```
                        fib(5)
                      /                 \
                fib(4)                 fib(3)
              /        \               /       \
         fib(3)       fib(2)          fib(2)   fib(1)
         /    \       /    \         /      \
  fib(2)   fib(1)   fib(1) fib(0) fib(1)  fib(0)
   /    \
fib(1) fib(0)
```

4. Optimize

Turn your crappy brute-force solution into something better!

Make sure that you **talk out loud** as you think (this is a good rule in general).

Interviewers do not like it when you sit silent for too long and just mull. Remember, they're evaluating your **thought process:** so make it super clear.

As you do more Leetcode/practice problems, you will get better and better at this. **Practice practice!**

4. Optimize

Our Problem:

We’re recalculating values!

For example: to find fib(4), our recursive method calculates the values of fib(3) and fib(2). But to find fib(3), we need the value of fib(2)! Thus, our method ends up calculating fib(2) twice!

Solution:

What if we stored the value of every result we calculate, and then use that result when calculating the result for larger numbers…

4. Optimize

Problem: Given a number n, print n-th Fibonacci Number.

```
public int fib(int n) {
    int f[] = new int[n+1];
    f[0] = 0;
    f[1] = 1;
    for (int i = 2; i <= n; i++) {
        f[i] = f[i-1] + f[i-2];
    }
    return f[n];
}
```

Storing results in an array allows us to access them in O(1) time!

This approach is known as **Dynamic Programming**

5. Walk Through

At this point, you should have a clear idea of what you're going to do.

Explain your planned algorithm to your interviewer at a **high level**. If you can, explain your approach as **comments in English** on the whiteboard/codepad

If you've been doing a good job of **thinking out loud,** then this should be really just a quick recap of what you've said so far.

6. Implement

Time to actually **write code.**

Writing code on a whiteboard is **hard**. You won’t have an IDE reminding you of syntax, and you’ve probably never done it before.

So practice this **often**.

Don’t fret tiny details of Java or other languages- but correct that if you have the time.

6. Implement

```
static int fib(int n) {
    int a = 0, b = 1, c;
    if (n == 0)
        return a;
    for (int i = 2; i <= n; i++) {
        c = a + b;
        a = b;
        b = c;
    }
    return b;
}
```

Updated Optimization: Since the current fibonacci value is the sum of the last two values, we only need to store the last two calculated values instead of storing a whole array!

7. Test

Test with Middle Case 2:

```
n = 9
Fib = 1, 1, 2, 3, 5, 8, 13, 21, 34
```

Resulting variable values after for loop ends:

```
a = 21, b = 34, c = 34
Return b = 34
```

Want another CEAOW-IT Example?

Check out **Problem 1** on the PDF.

Optional: CEAOW-IT Yourself!

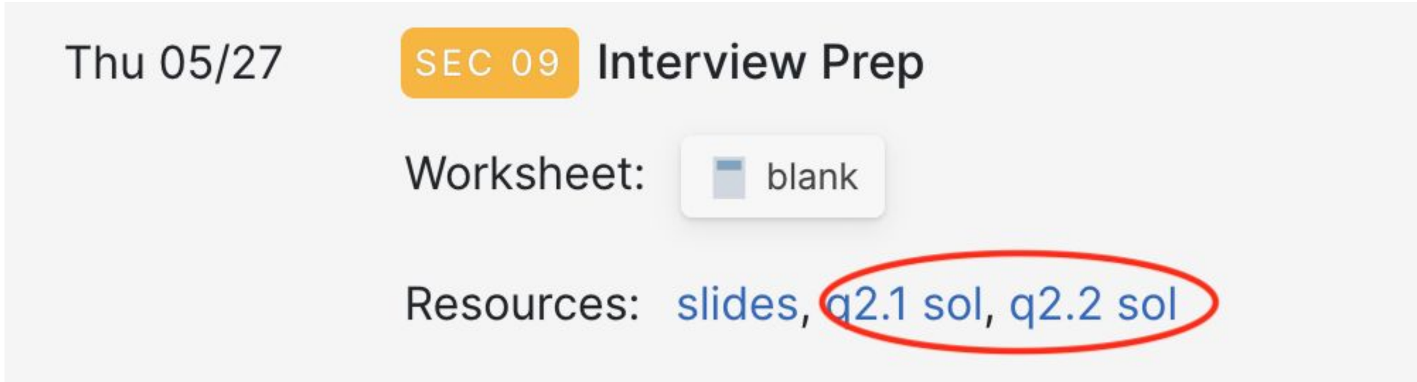
Pair Practice

- We’re going to put you in breakouts in **pairs** to practice the TEBOW-IT method.

- If you aren’t comfortable with this, it’s **totally OK** to leave now.

Pair Practice

- One partner will pick **Q2.1**, and the other will have **Q2.2**.
- For **10 minutes** one of you will attempt the TEBOW-IT method on the given problem while the other plays interviewer. Then you'll **switch** roles for another ten minutes!
- **Note**: The solution slides to each problem are on the website along with the worksheet. If you're acting as the interviewer, you should have the solution slides open for your **partner's** problem. If they get stuck, give them a **hint**!



Thu 05/27 SEC 09 Interview Prep

Worksheet: blank

Resources: slides, q2.1 sol, q2.2 sol