Technical Interviewing

How will you prepare?

1. DS&A
2. Pairboarding
3. Study: a/A Resource Library

Agenda

1. The Big Picture.
2. DS/Algos in the context of the interview pipeline.
3. Technical Interview Process
    a. Coding Challenge
    b. Whiteboarding

What are they looking for?

Big picture: Someone they want to work with

What are they looking for?

- Ability to work through a difficult problem and call on different tools
- Good design decision-making -- clean, readable, maintainable code
- Familiarity with a variety of “basics” that all software developers should know
- Good communication skills & fit with team members

Passing a technical interview is about *knowing the right stuff* and *communicating that you know it*

What's a "typical" pipeline?

Phone Screen
(20-45 minutes)

Coding Challenge or Project

Longer, Technical Phone (maybe Video) Screen
(45-60 minutes)

Virtual Onsite (3-6 hours)

Order varies, number of steps vary, level of organization varies

Why DS/Algorithms?

Algorithms/Data Structures- General Overview

Assessed via “whiteboarding” with a person

- ➔ The “classic” way to interview a software developer.
- ➔ Could be done on a physical whiteboard (pre-Covid) OR screen share software such as codeshare.io, Google doc, HackerRank, etc.

Or assessed via coding challenges

Bottom line: DS & A practice is ESSENTIAL to almost every facet of the job search

How to succeed in DS&A whiteboarding interview?

Six Steps:

- Understand (30%)
  - Ask clarifying questions (Do you understand the prompt?)
  - Setup Sample I/O's/Identify edge cases
- Plan (45%)
  - Identify approach(es) (What DSs combine to solve problem? Complexities?)
  - Pseudocode! (This is the most important step)
- Code (25%)
  - Implement your solution (CODE)
- Test and Analyze (100%)
  - Test your I/O's and revisit Big-O Notation

Algorithms/Data Structure- How to Prep

- Continue with the DS/Algos Course on AA Open
- Cracking the Coding Interview + Elements of Programming Interviews
- Leetcode + HackerRank
  - AlgoExpert + InterviewCake are also good, but they are NOT free
- Pramp: Practice *live* and *often (2 time per week)*
- Take notes, track, and analyze your own practice & study - what techniques are you using? When do you use them?
  - How are other people solving these problems?
  - Are they using techniques that you're not familiar with?

Coding Challenges

Coding Challenge Examples

- ➔ Build React to-do app
- ➔ Given hypothetical media company that wanted to track user streaming and billing, create ERP, design algorithm based on ERP to send notifications to people who are late on their payment, and query to return subscriptions.
- ➔ Given text file, with obstacle coordinates (numbers) and letters indicating directions (NSEW). Each movement need to check for obstacles, and output of function is max distance you can move at any time from starting position. (1 hour)
- ➔ Given array of page lines (on document), return information that gives each unique word frequency overall and corresponding sentence numbers.

Coding Challenge Examples

- ➔ Build web application
- ➔ Leaderboard using OOP
- ➔ Given API, create live updating hypothetical satellite altitude web page, create API endpoints over HTTP, return min/ max/ average altitude over last five minutes. Create function to do a health-check to return warning if satellite goes below a certain altitude for more than one minute. Write unit tests and Readme.

Coding Challenge Best Practices

- ➔ Do the BONUS features
- ➔ Manage your time well
- ➔ Ask recruiter if they have any recommendations about what you can study to prepare
- ➔ Review code and optimize before submitting if possible
  - ◆ If NOT possible due to time, continue working after submission and send with “thank you” email

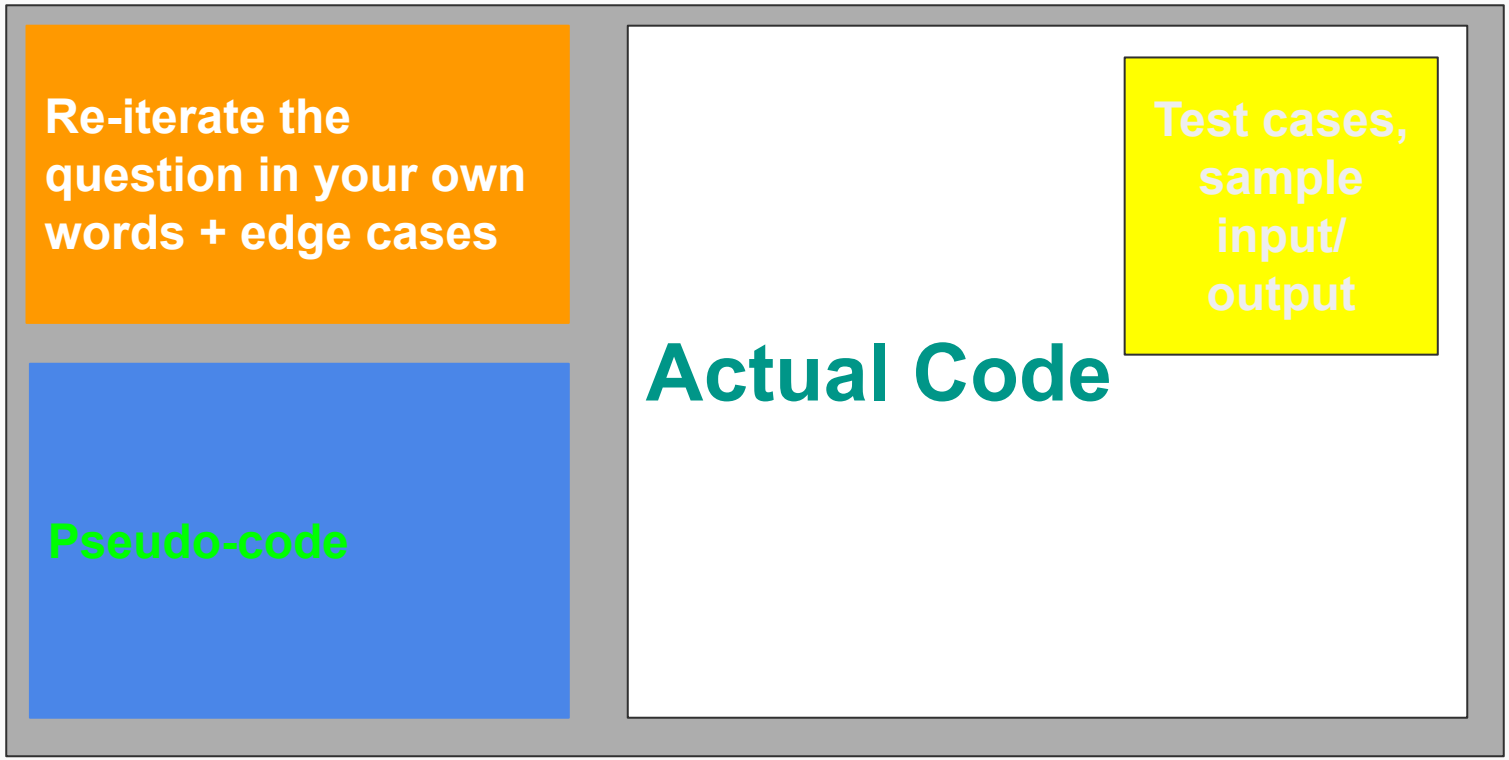
Whiteboarding

General Advice

A. Don’t rush and reiterate the question
    a. Guarantee understanding! Clarify assumptions.
B. Articulate your thoughts
    a. “Writing things down helps me keep track of my thoughts, would it be cool if I jotted things along the way as I explain?”
    b. Pseudocode
    c. Turn and talk

Understand + Plan + Code + Test and Analyze

Whiteboard Organization



Example Problem

Overview: For two strings to be isomorphic, all occurrences of a character in string `A` can be replaced with another character to get string `B`. The order of the characters must be preserved. There must be one-to-one mapping for every char of string A to every char of string B.

`paper` and `title` would return true. `egg` and `sad` would return false. `dgg` and `add` would return true.

```
isIsomorphic("egg", 'add'); // true
```

```
isIsomorphic("paper", 'title'); // true
```

```
isIsomorphic("kick", 'side'); // false
```

Understand

- Restate the question
- Ask clarifying questions
- **Example inputs and outputs** - clarify that what you expect to work actually does
- Talk/ask about edge cases - sooner rather than later

Plan: Brainstorm

- Brainstorm different approaches
  - “Here’s what I’m thinking… “
- This step comes right before you write your pseudo-code.
- Write out the different approaches in English
- Constant clarification
  - “How does this plan of action sound?”
    - Keeps interviewer engaged
    - Exemplifies collaborative tendency

Plan: Talk through differing approaches

- Does it work with your example inputs?
- Time and Space Complexity (Trade Offs)
- Pick the best one

Plan: Pseudocode

- ➔ Clear communication
- ➔ Ability to think through complex prompts
- ➔ Demonstrates good listening skills (assuming you didn't make any incorrect assumptions)
- ➔ Good place to talk more about edge cases
  - ◆ Shows that you are proactive
- ➔ Example: Good, bad, ugly

Write and Test Code

- Find bugs
- Do any last analysis of time complexity and space complexity

Recognizing mistakes in existing plan is crucial and absolutely fine.

Pairboarding Tips

- Be honest with your partner.
- Avoid positive body language response
- Give concise, constructive criticism! (You're trying to help each other get better.)
- Ask good questions and limit assumptions

Tech Trivia

Knowledge/Trivia- General Overview

"Do you have a baseline understanding of various important subjects in web development and **can you communicate your understanding well?**"

Knowledge/Trivia - Examples

1. What is the event loop? How does it work?
2. What happens when you type in ‘www.google.com’ and hit enter?
3. What’s the best and worst part about using Flask?
4. What is REST and what does it stand for?

Knowledge/Trivia - Examples

1. Talk about the 4 types of positions in CSS.
2. What is HTTPS, and how is it different from HTTP?
3. What's a ring buffer?
4. What's your favorite thing about your favorite language? What's your least favorite thing about your favorite language?
5. What's a load balancer?
6. What does responsive mean in terms of web pages?
7. What's a hashmap vs hash table?
8. What is XSS and CSRF? what are the differences between the two?
9. Tell me about the Event Loop in JS.

Knowledge/Trivia- How to answer well?

- Structure your responses:
  - Don't rush. Take a couple of moments to think.
  - Thesis statement: "The event loop is Javascript's solution to handling asynchronicity. It has three parts: an execution stack, a callback queue, and the Web Api. Here's an example..."
- Use an example to illustrate the idea:
  - Example should be small and show benefits of tool/concept.

Knowledge/Trivia- How to Prep

- ➔ Prepping:
  - ◆ Understand every single line of code you write, and know the curriculum.
  - ◆ Flashcards (ie. Enki; Brainscape) - If this works for you.
    - Emphasize *learning*, not just reading.
    - Retention and Performance
  - ◆ How do you learn best? The Resource Library is full of videos, texts, and interactive guides to continue cultivating your tech trivia

Much more to come during Tech Trivia workshop tomorrow

Questions

Demo Time!

The interview is over, what happens next?

- ➔ Always, ALWAYS send a follow-up thank you email to everyone who met with you
  - What if you don't get their email and only caught their first name?
    - Search the company on LinkedIn and filter by everyone with the first name.
      - Success: Follow the email address layout of whoever it was that set up your interview.
        - Success: GREAT
        - Unsuccess: Send a follow up to your contact, asking them to forward along your thank you email to your interviewer, ${NAME} AND connect with interviewer on LinkedIn - include a brief thank you here as well.
      - Unsuccess: Same back-up plan as above.

Wrap Up

Thank you's are not optional:

1. Ask HR to forward email to contact

   AND

2. Connect with interviewer on LinkedIn, including brief message.

- "Hi NAME, thank you again for taking the time to interview me this afternoon. I think I'm a perfect fit for ROLE and sincerely hope to be working alongside you soon. Please let me know if you have any other questions. I look forward to hearing from you! YOURNAME