Friyay 1

What will you have to do to get a job in development?

Interview Process

1. Phone Screen
2. Take-home
3. In-person Technical Interview (could be multiple)
4. Compatibility Interview
5. Offer

Interview Process

3. In-person Technical Interview (could be multiple)

Technical Interview Parts

1. Basic knowledge questions
2. Whiteboarding
3. Questions for interviewer

What is whiteboarding?

Whiteboarding

It is the process of answering coding related questions without a computer, out loud, while people ask you a ton of questions.

Why do they do this?

Because coding doesn't have a professional certificate, and coders are paid well, companies have to go through extra tests to make sure people can code.

How do I get good at it?

PRACTICE!!!

That's why we have algorithm/job interview Fridays

Problem Solving

Steps for breaking down a coding problems:

1. Understand the problem
2. Explore concrete examples
3. Break it down
4. Solve/Simplify
5. Look back and refactor

Understand the problem

Before writing ANY code, you should ask yourself or the interviewer some questions.

Understand the problem

- Can you restate the question in your own words?
- What are the inputs that go into the problem?
- What are the outputs of the problem?
- Can the outputs be determined from the input?
- How should I label the pieces of data that are part of the problem?

Example

Write a program that asks the user for two numerical inputs and adds them together.

What questions should you ask?

Restate Question

Implement addition

What inputs are going into the problem?

- Strings?
- Floats?
- Really, really big numbers?
- Negatives?

What should come out of the program?

- Integer
- Float
- String?

What should I label things?

- Naming variables is hard

Explore Concrete Examples

Concrete Examples

- Helps you understand the problem better
- Gives you a sanity check to make sure that something isn't too easy or hard in your thinking

Concrete Examples

- First, start simply. What's the simplest example you can think of?
- Then go on to more complicated examples.
- Then go to edge cases

Edge Cases

- What if you have empty inputs?
- What if you have invalid inputs?
- What if they try and input code?
- etc.

Break Down Problem

- These are the steps you write in comments about how you will solve the problem
- Can sometimes get hints from the interviewer
- Also lets you know what parts of the problem you're not confident in
- Should be done prior to writing code

Simplify

- Find the difficulty of what you're trying to do
- Ignore that difficulty for now
- Write a simple solution
- Then incorporate that problem back in

Refactor

- Can you check the results?
- Is there duplicated code?
- Does it make sense at a glance?
- Can you improve the performance of your solution?

Exercise:

- Take a user's input for a number, and then print out all numbers from 1 to that number. For any number divisible by 3, print 'fizz'. For any number divisible by 5, print 'buzz'. Any number divisible by 3 AND 5, print 'fizzbuzz'.

Exercise 2:

- Write a program that asks the user for a numerical input, 'n', and prints out the next 'n' numbers of the fibonacci sequence. (1, 1, 2, 3, 5, 8, 13, 21, 34...)