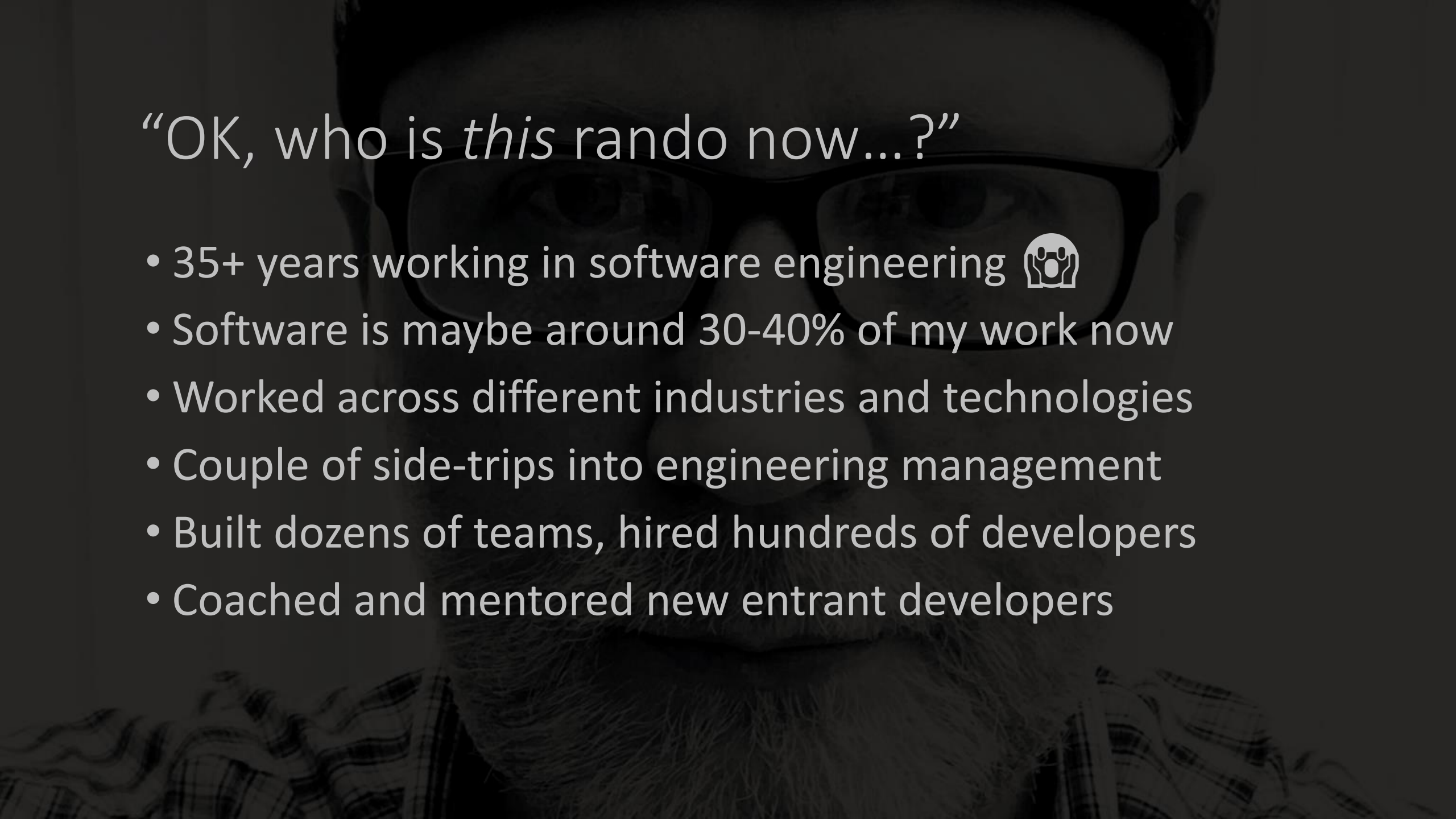
Technical Interviews, Tests, and Technical Hiring

Mike Ritchie
mike@evolved.dev

How it works, what to expect, and why companies do it.

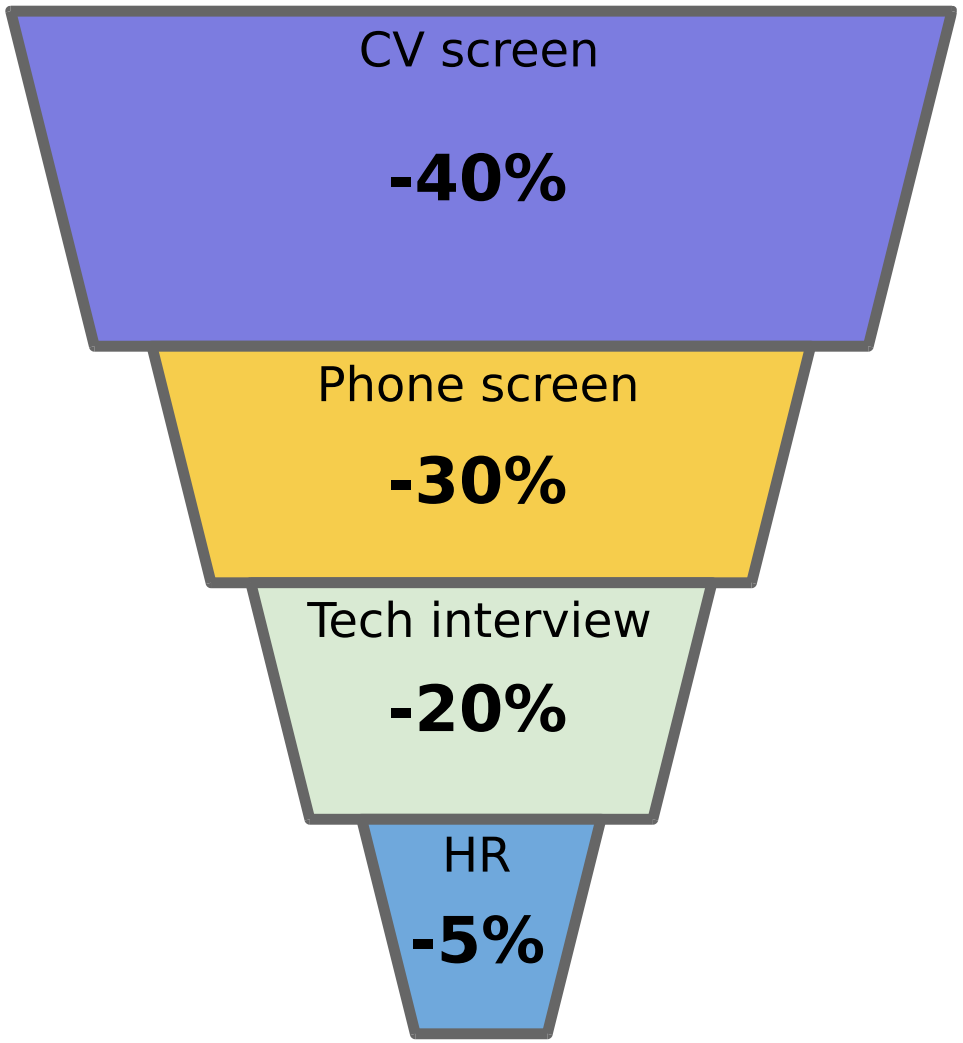


“OK, who is *this* rando now…?”

- 35+ years working in software engineering 🫣
- Software is maybe around 30-40% of my work now
- Worked across different industries and technologies
- Couple of side-trips into engineering management
- Built dozens of teams, hired hundreds of developers
- Coached and mentored new entrant developers

Technical Recruitment

Process, participants, and the early stages of selection.



The hiring funnel

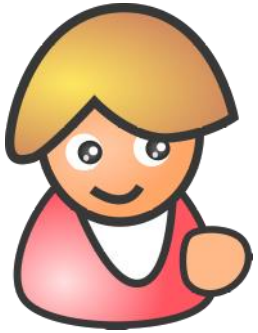
- Most companies operate some variant of this model
- The cost of interviewing and hiring is a factor in keeping the face-to-face until last
- There’s an alarming truth implied by the shape of this funnel…

The cast of characters

Reginald the recruiter

You the amazing candidate

Henry the hiring manager



Theresa the tech interviewer

Different perspectives, but some things make all of us happy…

Consistency	Words and actions match
Clear narrative	Your story is credible and authentic
In the driving seat	You're setting the pace for your own development
Positivity	You're upbeat and engaged in the interview

...or unhappy

Disinterest and disengagement

Kills 99.9% of job applications.

Stone.

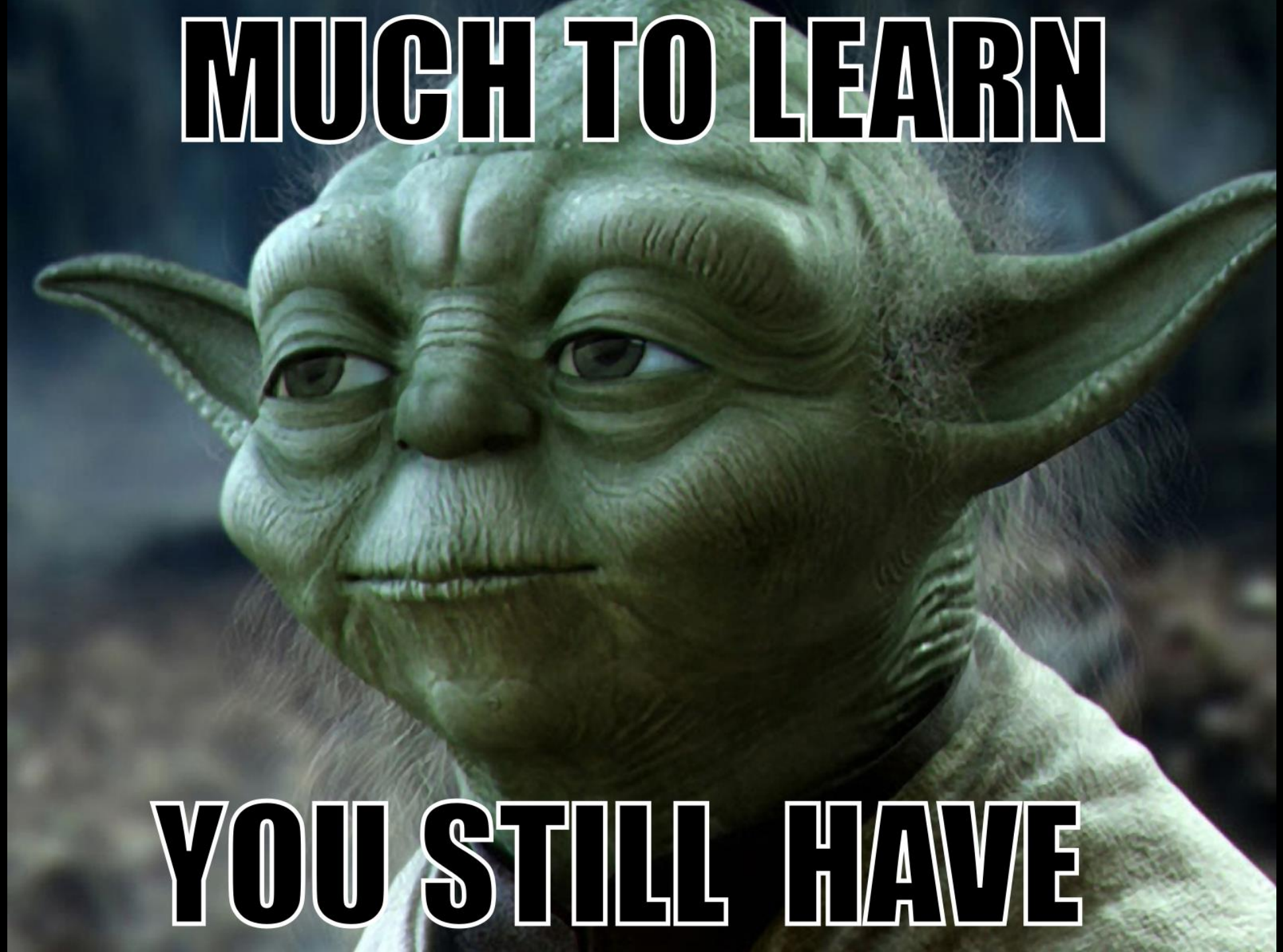
Dead.

Kevlin Henney, in the preface to Pete Goodliffe's book "Becoming a Better Programmer"

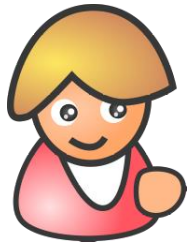
All too often, programmers are divided into average programmers and rockstar or ninja programmers. Where there's a rockstar, there's a trashed codebase with broken classes and spaced-out control flow. Where there's a ninja, there's mysterious bugs and build problems that occur in the middle of the night.

Where there's an average, there's a distribution. In the long term, what matters is less **where on the distribution someone is** than where they are **headed.**

If you want to divide programmers into two groups, there are programmers who get better and programmers who don't. **You care about the first group.**


MUCH TO LEARN
YOU STILL HAVE

Theresa is looking for two key things



Trajectory

You're a "getting better" person, and you'll keep working on your skills.

You'll add to the team, and be someone that the team wants to work with.

Coachability

You'll be receptive to coaching, an active listener.

You will follow through on suggestions and advice from experienced engineers.

Your CV is more than your CV

Getting the most out of your GitHub profile.

mike ★

50ohm

Follow

Modern C++ for embedded, SysML when things get complex, and Python when I just need to Get Shit Done™

67 followers · 4 following

@EvolvedHQ

https://evolved.dev/

@Evolved_HQ

Overview | Repositories 2 | Projects | Packages | Stars 62

Pinned

EvolvedHQ/cookiecutter-kata-catch Public

A cookiecutter template for deliberate TDD practice in C++ with Catch! and Trompeloeil

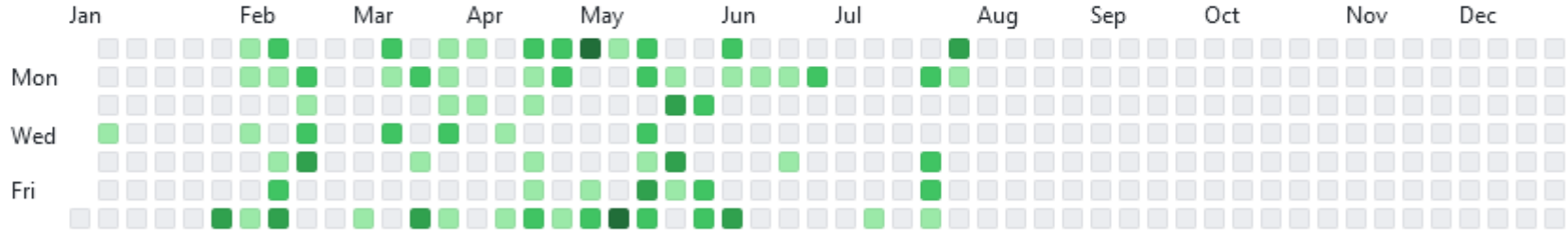
C++ ☆ 2

EvolvedHQ/cookiecutter-cpp-clion Public

An unashamedly CLion-friendly cookiecutter template for C++2a, TDD with Catch2 and code coverage.

C++ ☆ 2

225 contributions in 2022



Learn how we count contributions

Less More

Contribution activity

2022

2021

2020

2019

August 2022

2 contributions in private repositories — Aug 1

mike ★

50ohm

Follow

Modern C++ for embedded, SysML when things get complex, and Python when I just need to Get Shit Done™

67 followers · 4 following

@EvolvedHQ

https://evolved.dev/

@Evolved_HQ

Overview | Repositories 2 | Projects | Packages | Stars 62

Pinned

EvolvedH

A cookiecutter
Trompeloeil

C++ ☆

EvolvedHQ/cookiecutter-cpp-clion Public

An unashamedly CLion-friendly cookiecutter template for C++2a, TDD with Catch2 and code coverage.

C++ ☆ 2

Try to populate this as much as you can – it gives your profile a more "looked after" feel

225 contribu

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

Mon

Wed

Fri

Learn how we count contributions

Less More

Contribution activity

2022

2021

2020

2019

August 2022

2 contributions in private repositories

Aug 1

Overview | Repositories 2 | Projects | Packages | Stars 62

mike ★

50ohm

Follow

Modern C++ for embedded, SysML when things get complex, and Python when I just need to Get Shit Done™

67 followers · 4 following

@EvolvedHQ

https://evolved.dev/

@Evolved_HQ

Pinned

EvolvedHQ/**cookiecutter-kata-catch** Public

A cookiecutter template for deliberate TDD practice in C++ with Catch! and Trompeloeil

C++ ☆ 2

EvolvedHQ/**cookiecutter-cpp-clion** Public

An unashamedly CLion-friendly cookiecutter template for C++2a, TDD with Catch2 and code coverage.

C++ ☆ 2

225 contributions in 2022

Jan Feb Sep Oct Nov Dec

Mon

Wed

Fri

Learn how we count contributio

Less More

Customise the repositories that you want to showcase by pinning them to your profile front page

Contribution activity

2022

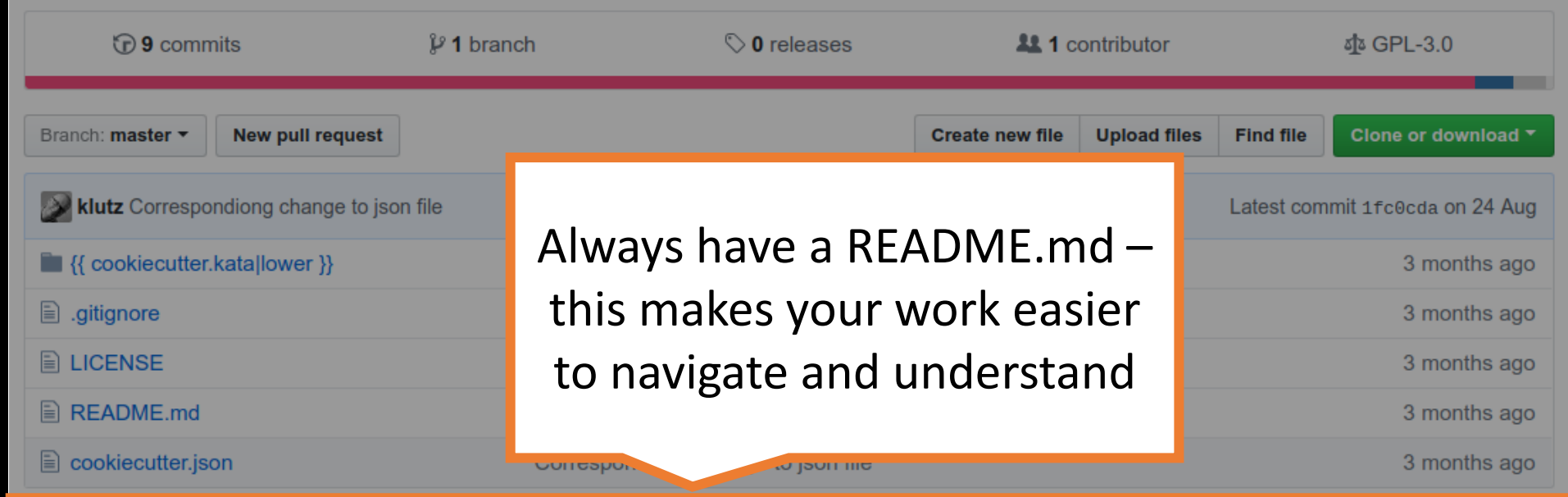
2021

2020

2019

August 2022

2 contributions in private repositories Aug 1



Always have a README.md – this makes your work easier to navigate and understand

README.md

Template for TDD Code Katas in C++

Hey there. This is a cookiecutter template for a simple TDD code kata using Catch! and Trompeloeil. It's intended to give you a repeatable way of very quickly getting started for a "deliberate practice" session with C++.

Features of this template

This generates a project for doing a test-driven code kata in C++.

- Includes Catch! and Trompeloeil libraries
- Generates a header, "production" source file and an empty test
- Generates a CMake build which will work on most platforms
- Has some convenience targets for generating etags and running unit tests

Screening calls

Phone, Zoom, Skype…different platforms, same goal.


BBC
NEWS
LONDON
BERLIN
PARIS
ROME
JERUSALEM
CAIRO
NAIROBI
JOHANNESBURG

Screening question formats

Experience-based

- “Tell me what you did on <project>”
- “How did you find switching to <language>?”

Knowledge-based

- Language, core library APIs
- Programming fundamentals
- Tools, libraries, frameworks
- Likely connected to:
  - The role being filled
  - Your past experience

Screening call tips

Be friendly – remember it's a filter

Keep your answers concise

Ask for clarification when you need to

If you don't know, just say so

Oh yeah, and don't do this….

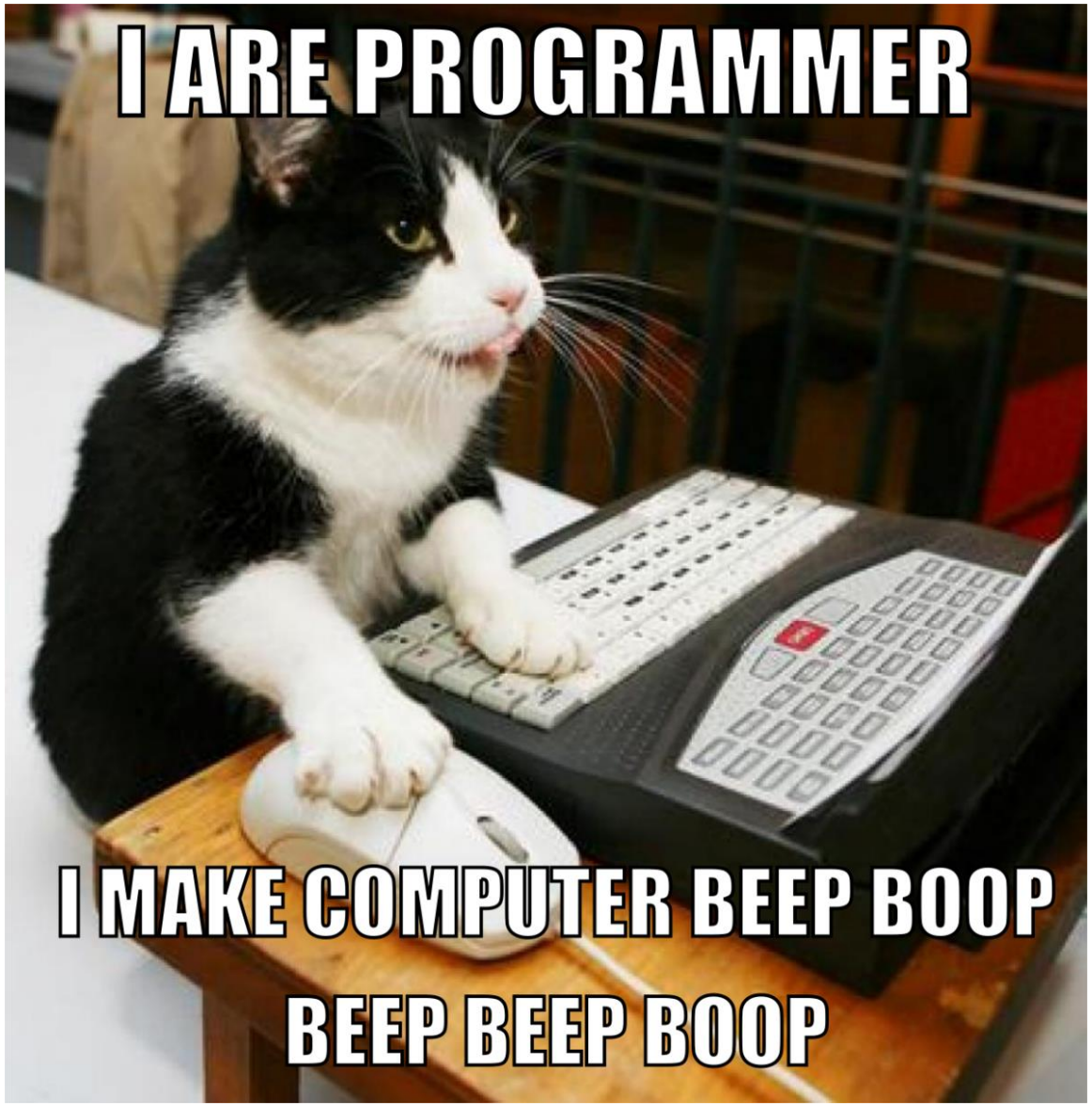
Google

how do I reverse a string in

how do i reverse a string in **python**
how do i reverse a string in **java**
how do i reverse a string in **c++**

Technical Interviews

Buckle up - this is the main event, and there's a lot of it.



Technical interview formats, rated by pandas

Technical Q&A

Pair programming

Whiteboard design

Online code tests

Homework projects

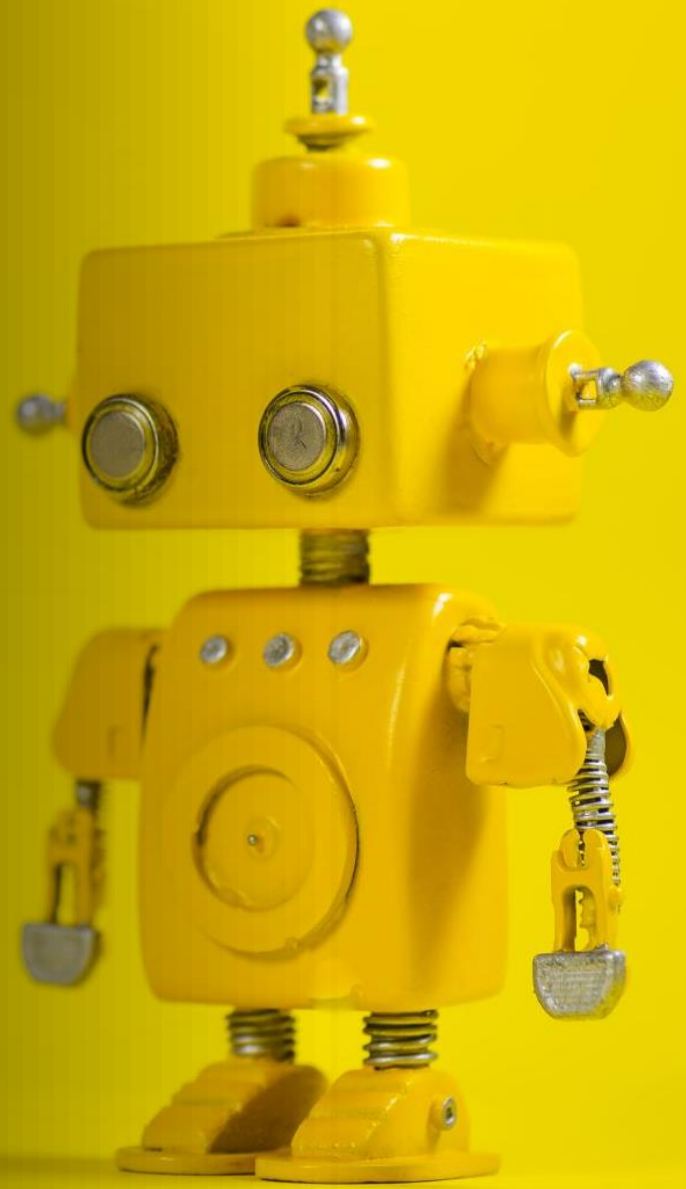
Bug hunts

Nerves

- Focus on the problem, not the interview
- Think out loud, and sketch your ideas
- Discuss with the interviewer, always
- Don’t over-think - orient yourself, then begin
- And remember….

…your tech interviewer f**ks up too

Pair programming interviews



Pairing platforms

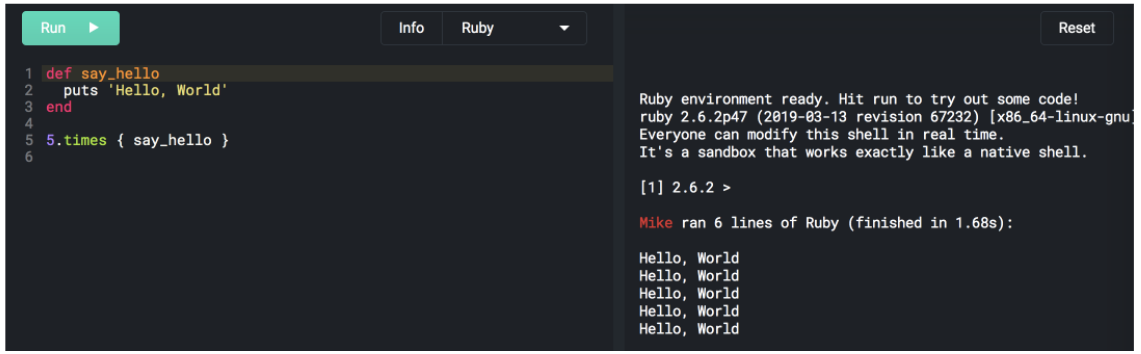
https://www.jetbrains.com/code-with-me/

https://code.visualstudio.com/

CoderPad

Hire better candidates faster.

Leave the whiteboard behind with CoderPad's intuitive, live environment.



https://coderpad.io/

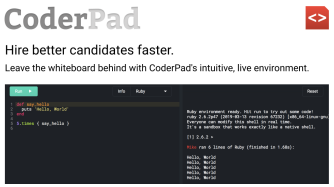
Pairing practice

These go much better if you have some familiarity with the tools.

Also aim for fluency in pairing – practice as much as you can with other students or instructors.

This is also great practice for learning on the job by pairing with experienced devs.

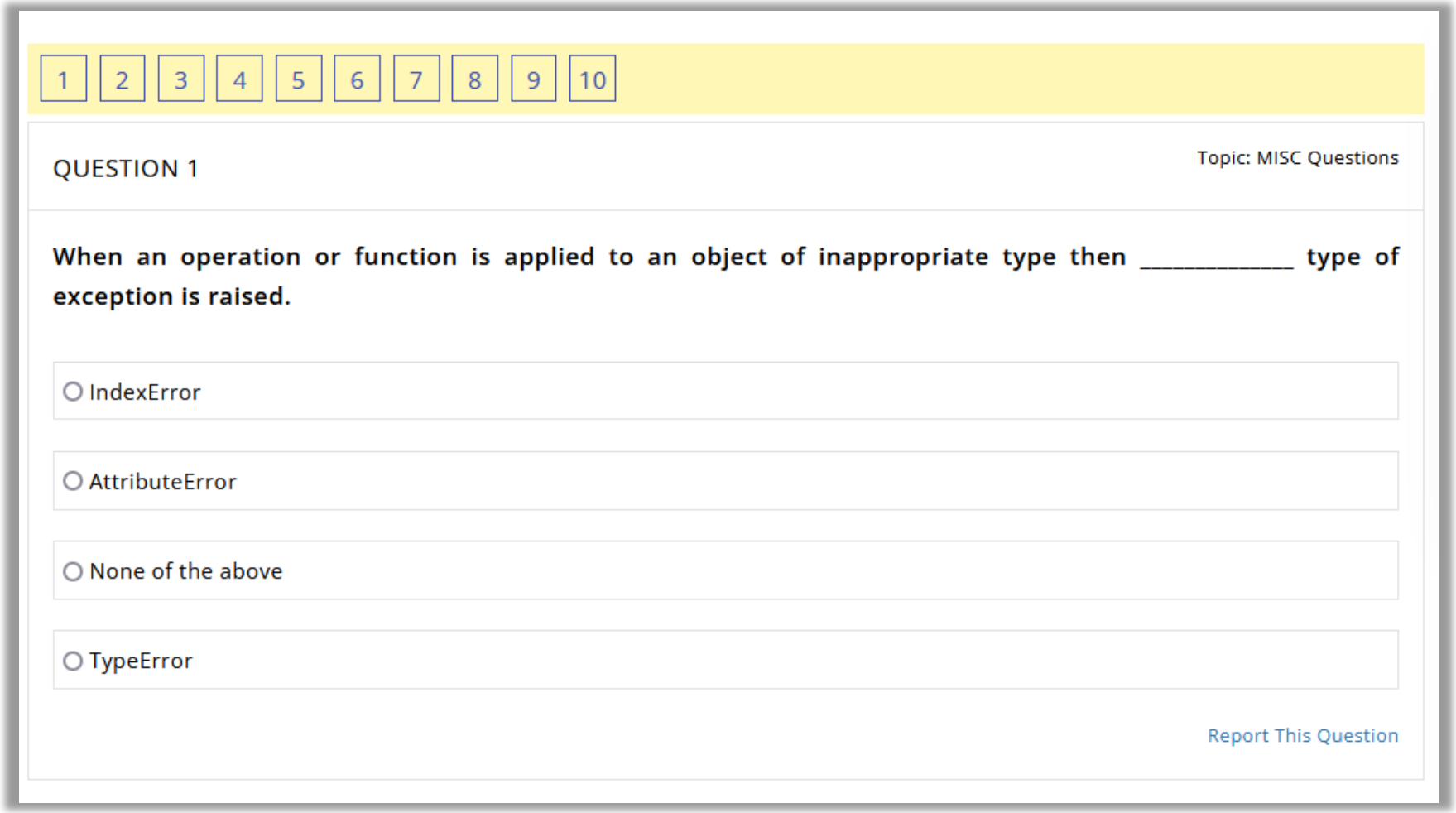
- ✓Ask in advance what platform will be used
- ✓Baby steps is the key, working in tiny increments with tests
- ✓Google “code kata” for practice problems



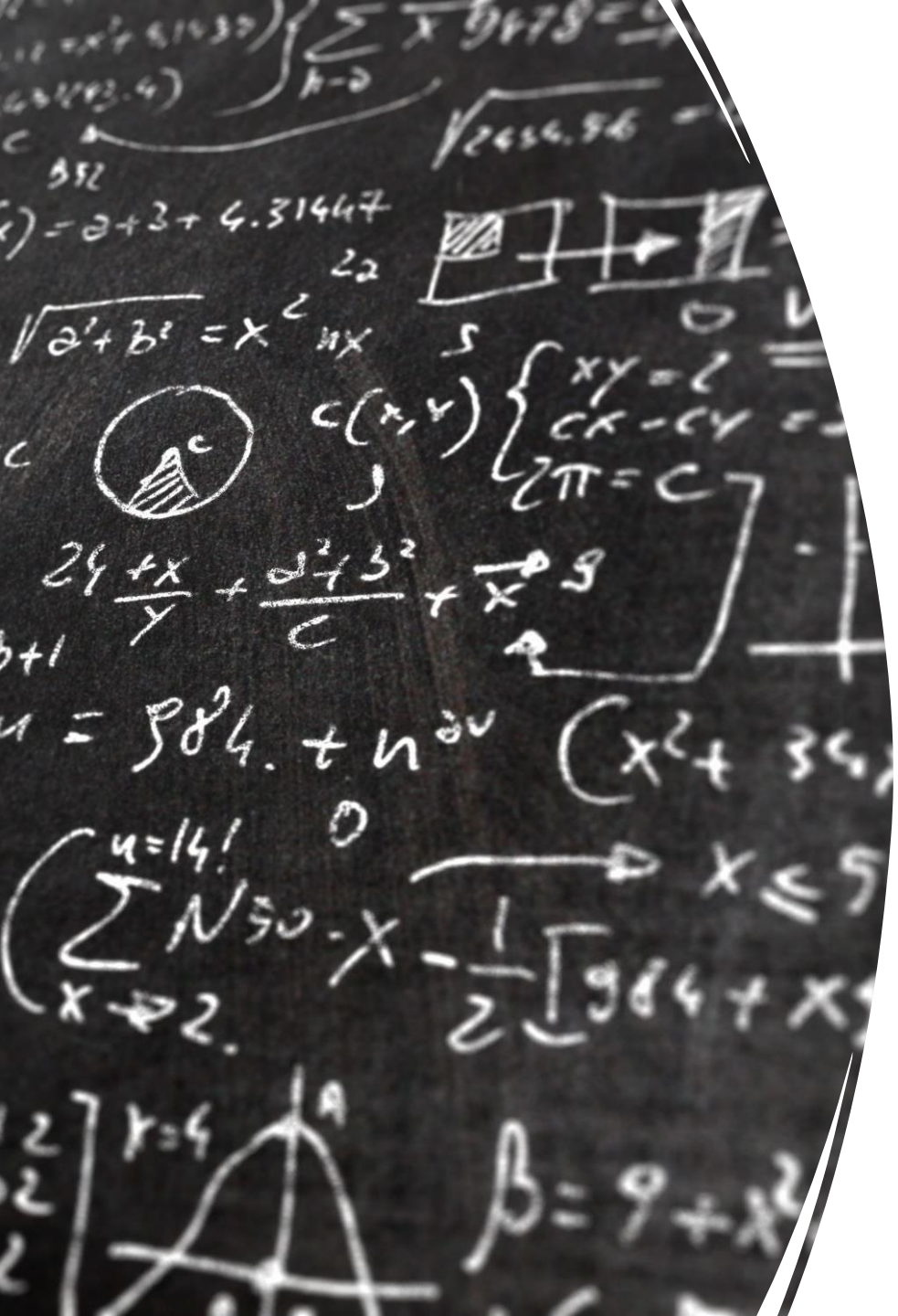
Online tests

- These come in different formats, some harder than others
- Some are about testing knowledge – e.g. of a language
- Others are more self-contained with a limited “make this existing solution work”
- Some require you to develop whole problem solutions
- Most platforms have a try-out sample for candidates

Multiple choice knowledge-based questions



Small-scale coding problems



Hackerrank, Codility, and many clones

- These aren’t great for candidate experience
- Will typically test programming fundamentals
- Usually need algorithmic & numeric thinking
- You can pick your choice of language
- Bog-basic editor and timed countdown for added freak-out

HackerRankX

https://www.hackerrank.com/

codility

https://www.codility.com/

This is a demo task.

Write a function:

```
int solution(vector<int> &A);
```

that, given an array A of N integers, returns the smallest positive integer (greater than 0) that does not occur in A.

For example, given A = [1, 3, 6, 4, 1, 2], the function should return 5.

Given A = [1, 2, 3], the function should return 4.

Given A = [−1, −3], the function should return 1.

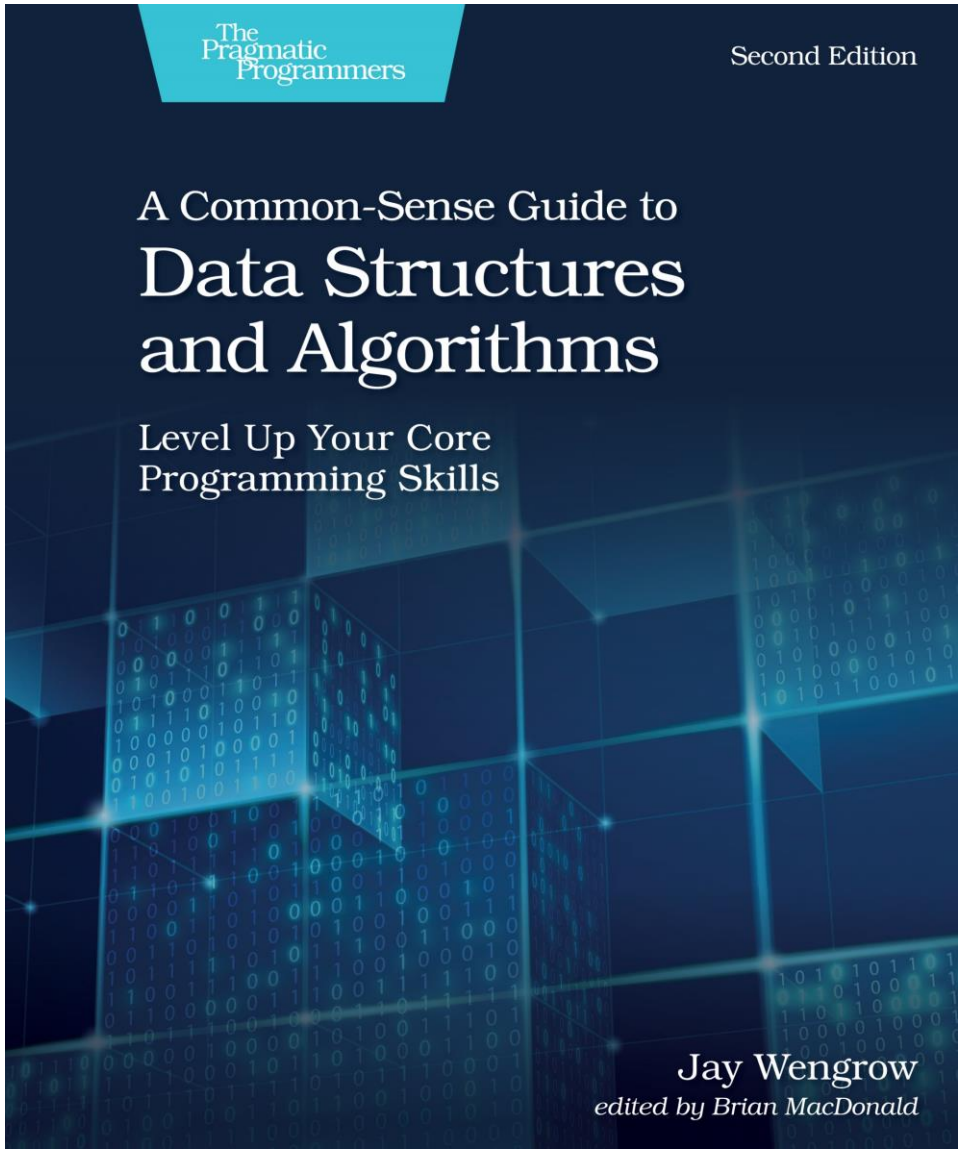
Write an **efficient** algorithm for the following assumptions:

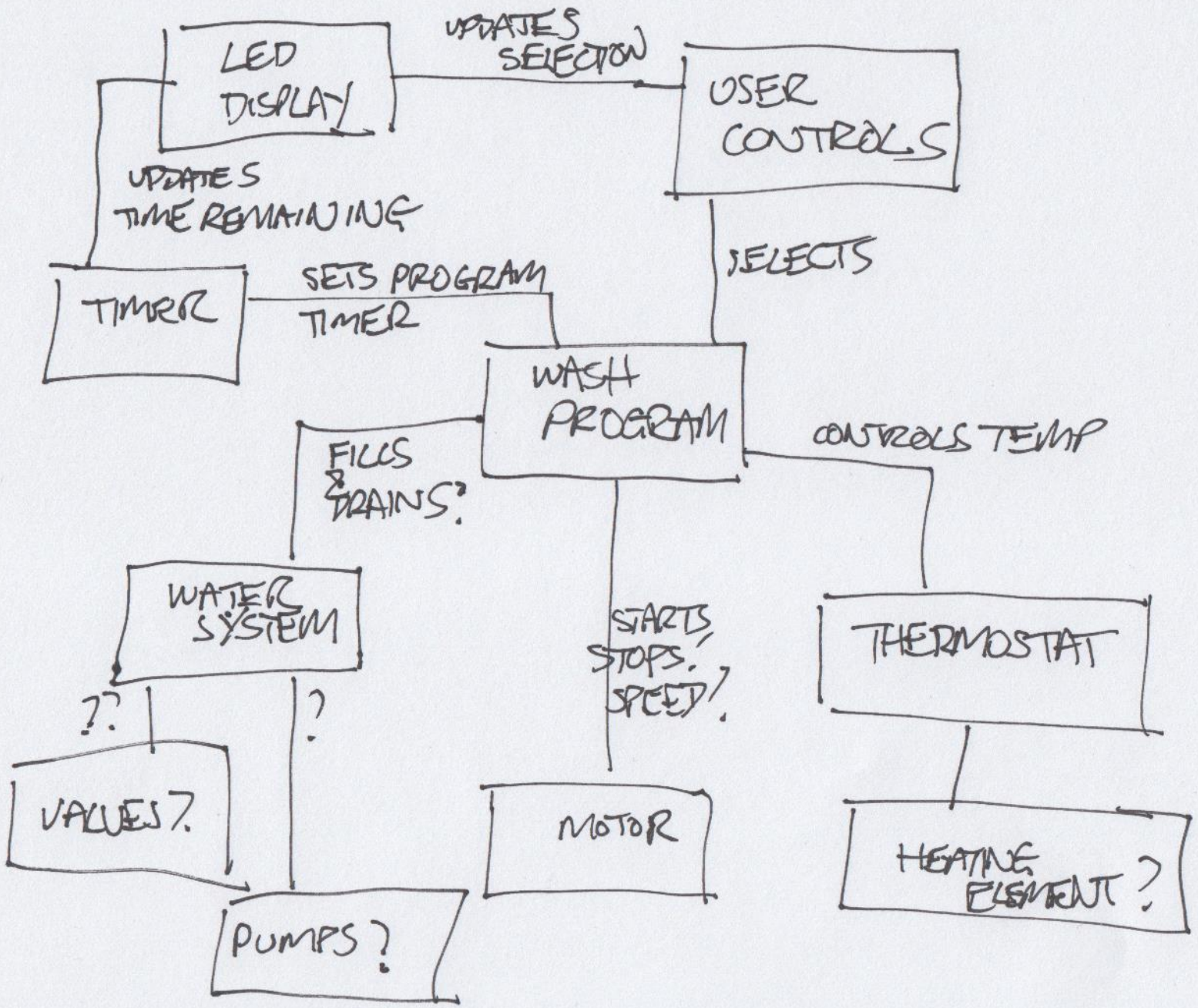
- N is an integer within the range [1..100,000];
- each element of array A is an integer within the range [−1,000,000..1,000,000].


```
1
2  int solution(vector<int> &A) {
3      // write your code in C++14 (g++ 6.2.0)
4  }
5
```

Online code tests practice and prep

- Get familiar with the major platforms – sign up for solo practice
- Learn some data structures and algorithms basics
- There are some great resources, from books to videos, and much of it free
- Practice this type of problem as a change of pace from your usual focus
- They can actually be fun (when nobody is watching…)





Whiteboard design

Quite common, often used when an interviewer wants to run with a technical theme, but not at code level.

You're given a problem and asked to sketch out a solution. Heavy emphasis on **sketch**.

Might be a generic problem, or something specific to the company.

For example, a bank might ask you to model bank accounts, transactions, customers, and debit/credit cards.

Engage the interviewer as if they were a colleague.

Practice, tips, and prep

You can achieve a pretty good level of fluency by practice with this format.

Sketching design is a great habit to form if you're going to be working in collaborative teams anyway – it happens **a lot** in software development teams.

Try sketching out a design for problems you are already familiar with or pick some familiar business domains.

"Classes, responsibilities, collaborators"

- What it is
- What it does
- What it works with

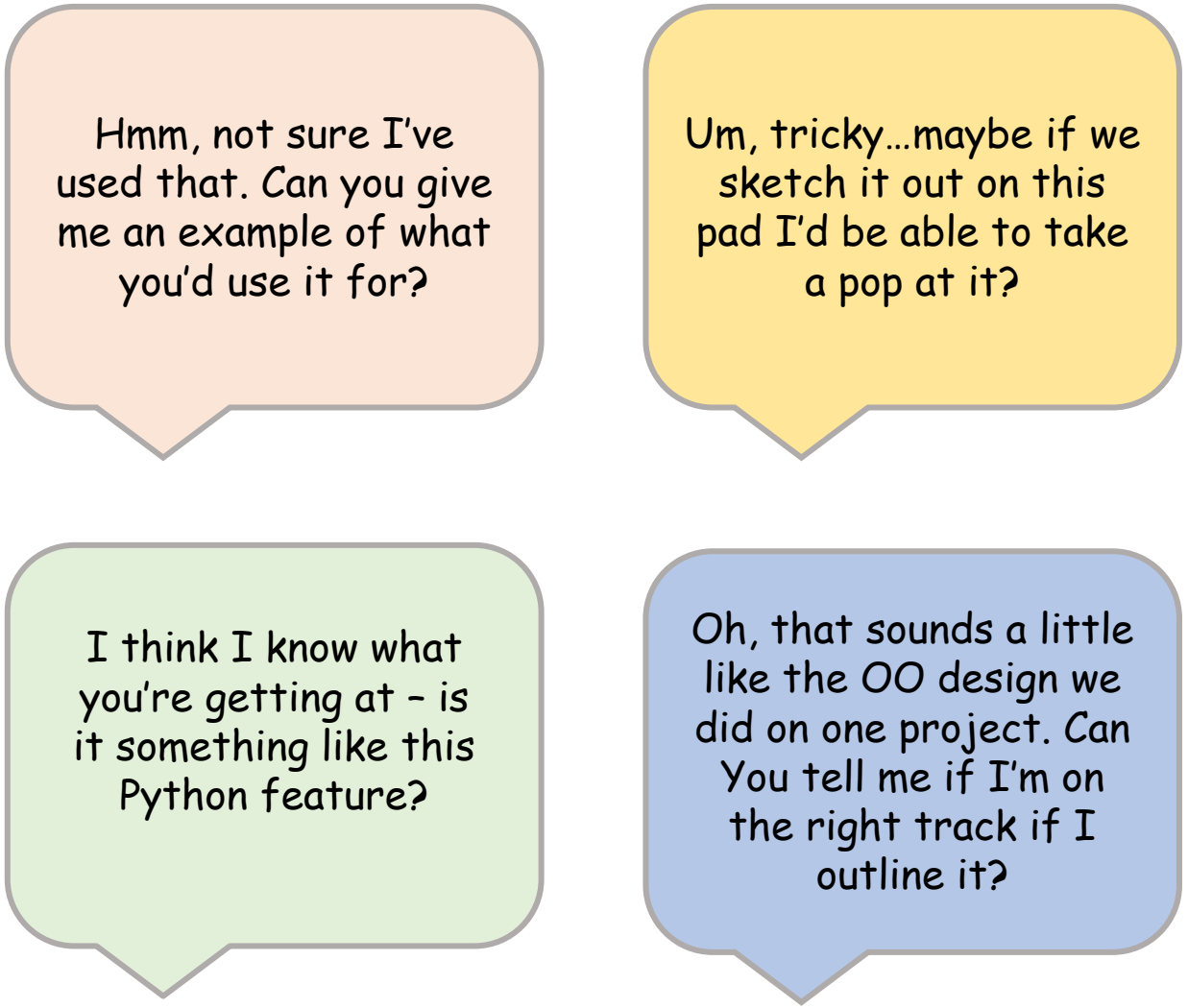
Broad sketch first to get a feel for the problem space, then deep-dive into details

Look for easy wins first (hint: find the nouns in a problem)

Q&A interviews

Your challenge is to turn one-way traffic into a discussion.

Aim to get the questioner more invested in their own question, supplementing, narrating, and expanding on the original question they asked.



Q&A practice and prep

It’s hard to predict what will crop up in such a free-form Q&A discussion, but the technique is as important as the content.

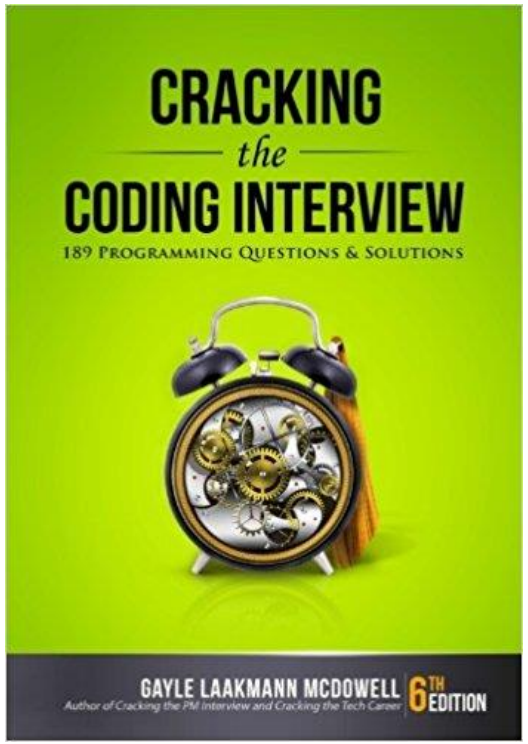
Use Google for technical questions!

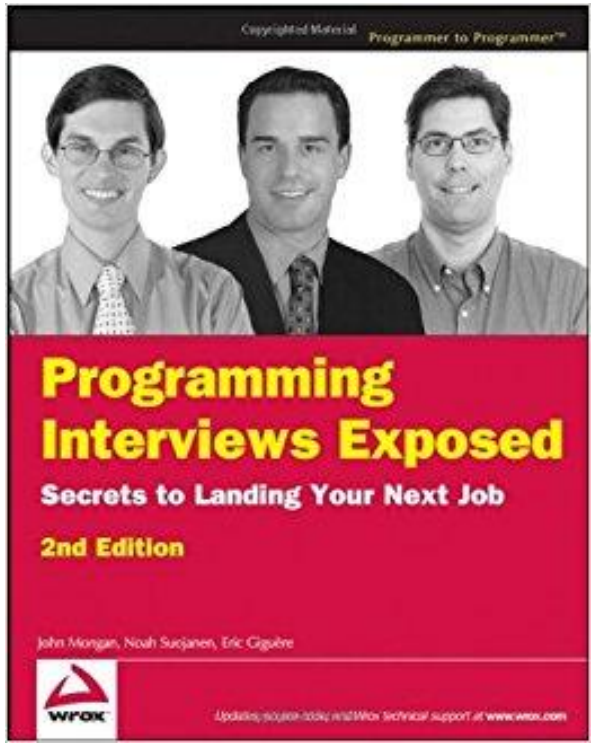
Use the content on your CV as guide for the things you should be able to talk about

Try to have an initial response rather than a fully rehearsed answer for CV questions

It helps a lot to speak your answers out loud

Books for technical interviews





Gayle Laakmann McDowell’s book is the go-to source for tech interview prep

That’s it!

Thanks for listening, and stay in touch.

mike@evolved.dev
https://www.linkedin.com/in/m-w-ritchie/
…or direct message in the CodeClan slack