Above & Beyond CS (ABCS)

Coding Interview Workshop Series

Workshop 6

Handling Mistakes

What have we seen so far?

Before coding

...the first 5'

1. Communicate Proactively ✓
2. Design Your Algorithm ✓
3. Work the Clock ✓

During coding

...the next 10'

4. Writing code at the whiteboard ✓
5. Talk through your code / solution ✓
6. Handling mistakes

"After" coding

...the last 2-3'

7. Test your code
8. Ask questions!

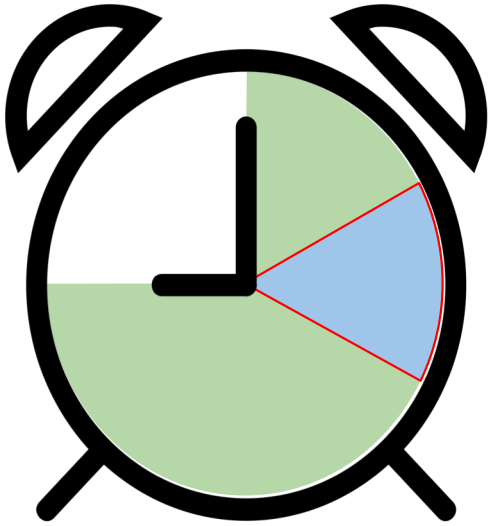
While you're coding...

4. Write code at the whiteboard
5. Talk through your code/solution
6. Handling mistakes

6. Handling Mistakes

Handling Mistakes

What could go wrong?



- Real life development is iterative. No one builds an app from scratch that runs without a bug.
- Your bound to make mistakes in your interview too – that's okay! Just be sure to catch them.
- The trick to doing well is staying calm and maintaining your composure

We All Make Mistakes

Let's get that out of the way!

- Interviewing, technical or otherwise, is stressful.
- Bugs are an almost inevitable part of coding!
  - Especially when we're problem solving in real time while also programming, it's very common to make mistakes. This is okay!
- Your interviewer is more interested in what you do with mistakes.
  - Do you catch them?
  - Can you fix them?
  - Can you get back on track?

How to Recover from Mistakes

Whatever you do, don't get flustered or down on yourself!

- Trace the code you've written with an example. Pretend it's someone else's code and that you're tasked with finding bugs in it.
  - By tracing the code you wrote, not what you intended to write, you'll catch mistakes
- Acknowledge the mistake, explain how to fix it, and implement the correction.
- If your interviewer spots an error and you can't find it, it's okay to talk through assumptions or ask for a hint. Listen carefully!

Remember: Ask Questions, Not for Hints

If you're stuck, it's okay to say you don't know.

- Don't try to appear like you know something when you don't
  - Instead say, "I'm not sure, but I'd guess ____ here because ____."
  - The "because" is important – it's an opportunity to rule out other options with poor tradeoffs, or by pulling in other examples from other languages or problems
  - After all, this is an opportunity to demonstrate how you tackle hard problems
- Ask questions, not for hints.
  - Questions aren't just for the first 5 minutes of the interview
  - It's okay to admit you're stumped; ask a question that helps identify the path forward
  - It's never a good idea to ask, "Can you give me a hint?"

Some Styling Mistakes to Avoid

Write code that other team members can pick up from where you left off.

- Give descriptive names to variables, functions, etc.
- Randomly mixing coding standards signals sloppiness
  - Use consistent naming conventions
  - Team Richard, or Team Winnie? (i.e. tabs or spaces?)
  - If you put braces on the same line, don't later put them in a new line
- Using defensive coding, such as NULL checks and special cases
  - This leads to more complicated code that is harder to understand and debug

Thank you!

Workshop 7
[INSERT REGION DATE/TIME]

Complete Pre-Work

- ✓ Review screencast
- ✓ Solve HackerRank problems
- ✓ Be prepared to walkthrough your submitted code