facebook

Above & Beyond CS (ABCS)

Coding Interview Workshop Series

Workshop 7

Testing Your Code

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. Increase Your Coding Speed
9. Tackling Imposter Syndrome

After you’re coding…

7. Testing Your Code
8. Increase Your Coding Speed
9. Tackle Imposter Syndrome

7. Testing Your Code

I'M DONE NOW
WOOHOO
makeameme.org

I'M DONE NOW
FALSE
WOOHOO
makeameme.org

Testing Your Code

Not testing your code could be a deal-breaker.

- Running your algorithm with a few examples will allow you to catch bugs and address them
- You may come up with an original algorithm your interviewer hadn’t thought of; testing will demonstrate that your solution works

Test, Test, Test!

You've finished writing your code. You're finished, right? WRONG!

- Failing to test your code is quick way to undermine yourself in your technical interview.
  - It's very common for code to fail in some specific test cases
  - After all, would you ever write code without running or testing it? I hope not!
- It's your job to read your interviewer's mind.
  - Don't make them do the intellectual heavy lifting
  - It's a huge plus if you write tests for your code without being prompted

What to Test For

Come up with small test cases and step through the code.

- **General Cases**: test the initial / normal case first
  - Note: be sure to handle all possible inputs!
- **Extreme Cases**: 0, negative, null, maximums, minimums, etc.
- **User error**: what happens if the user passes in a null or negative value?

Questions?

Let’s try it out!