Optimal Timelining

Data structures & Algorithms Interview

Hi, Welcome! Lets start at 6:35 PM
Please mute your audio, but feel free to unmute to interrupt me at any point
You can keep your video on/off, your choice

FastFunc

Agenda

- About this presentation & me
- Patterns of optimal & non-optimal Interviews
- Sequencing the 45-minute DS & algorithms interview
- Assessing yourself
- Iterating & Improving
- Questions for me

About this presentation

45 Min DS/Algo Interview

FastFunc

About me

- 10+ years Software Engineering
- FAANG++ interviews & offers. Hired engineers in my teams on several occasions.
- Part-time Interview instructor & coach
- A passion for optimizing everything around me for maximum fun & profit
- Sharing ☝️

FastFunc ⟨/⟩

Before we begin

- The average data structures & algorithms interview
  - F(X) , 45 minutes

- The work you have to put in
  - Problems & Practice
  - Not anxious about coding

- This is not a magic trick.
  - Repeatability often gets you this result naturally; I’ve encoded it into a set of steps

FastFunc

Non-optimal patterns

- No time-bound process
- Anxious and wants to start coding ASAP!
- Solutionism / pattern matching before exploring the problem space
- Unsure how to transition across stages. Unsure about stages
- Apologetic & asking for permission
- Looking for clues by trying to read the interviewers face

- Interviewer feels they met a job seeker

Optimal Patterns

- Aware of the clock
- Sets up conversational hooks to lead and direct the interview
- Senior engineering mindset : Questions & data to determine direction
- Extremely high fidelity conversation, pounces on new information
- Humble & decisive
- Coding is the finishing lap

- Interviewer feels they met an articulate problem solver
  - Parallels between Hollywood audition & technical interviewing
  - “Bryan Cranston’s advice to aspiring actors” -> Youtube video

The timeline again

45 Min DS/Algo Interview

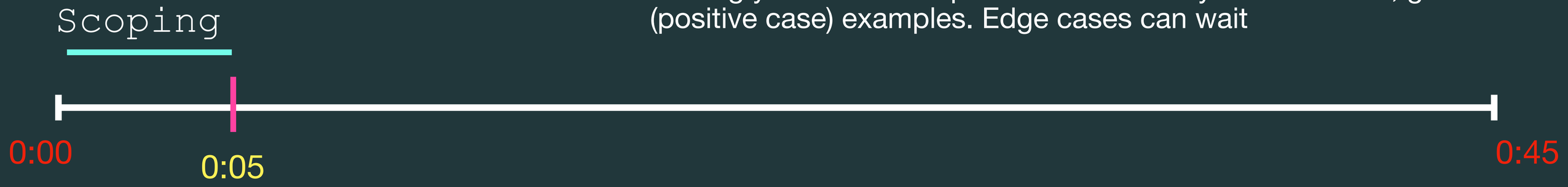
Pacing: Medium -> Slow -> Fast

Adapting this timeline

FastFunc

Scoping

1. Treat the problem as a “black box” : What are the inputs & outputs?
2. Don’t be tempted to entertain any solutions/patterns
3. Pay close attention to any examples given to you -break them down meticulously
4. Making your own examples is an art form: you want small, good (positive case) examples. Edge cases can wait



*Goal: Get as much information as you can about **HOW** the solution **behaves** under different inputs & outputs*

FastFunc

Brute force

1. Arrive at a brute force solution using simple constructs like loops (double, triple), exhaustive searching
2. Spell out the complexity. Declare that it should be improved
3. No need to code the solution or even whiteboard it

*Goal: Deliver a bad but working proposal with complexity. This is a **quick win**!*

FastFunc

Optimal Solution

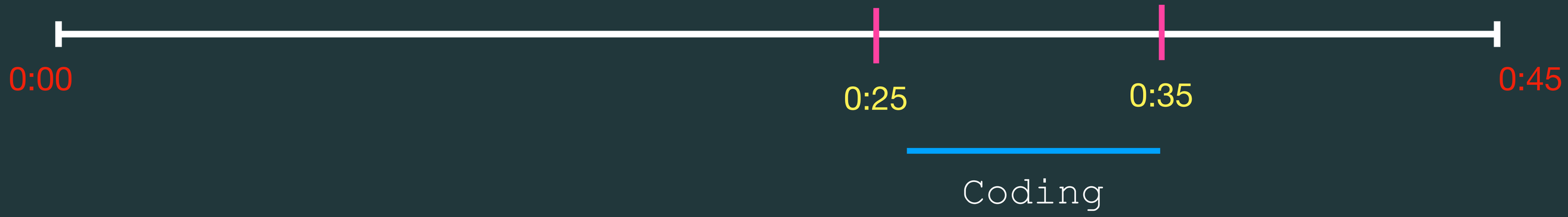
1. This is the phase where the insight arrives.
2. Use everything you've learnt so far to make *small, incremental good decisions*. Don't reach too far. Ponder, be slow, but talk aloud. Ask yourself questions aloud
3. Spell out the complexity - get the data if this is what they want -> transition
4. Aggressively narrow down on the core insight and get the pseudocode down. Can you visualize the code?

Goal: Get an optimal solution with complexity stated and pseudocode

FastFunc

Coding

1. This is the last lap, you should be happy when you get here (are you feeling this?)
2. Feels like copy-paste. Coding feels fast, top to bottom done very quick, you glance for bugs and find minor typos, syntax issues (did this happen?)
3. “OK lets check if this works” -> transition



*Goal: **Super smooth**, coding experience*

FastFunc

Walkthrough

1. Unit test mindset. Hit every line
2. 2/3 examples. First one done slowly. Speed up consecutive ones. Talking through every line
3. Spell out complexity of any language specific calls. They should not change the initial stated optimal complexity!

Goal: ‘unit test every line’

FastFunc

Buffer

1. You always have this space if you need it for finishing the problem
2. How to leave with an interesting impression? Do you know something about the company or interviewer?
3. Do you have a unique POV about a product, or a blog post you read
4. Ask many questions. There are some questions that are not valuable

Goal: Conclude on a pleasant, happy note. If you enjoyed the interview, tell them about it!

FastFunc

Additional variables

- Communication
  - Introverted vs Extroverted
- Managing the whiteboard/editor
  - You should have a ‘terminal view’ of the whiteboard in your mind
- Managing markers, eraser
  - I once met someone who carries their own markers and eraser
- Managing the clock
  - Wear a stopwatch, use dialogue
- Using language specific abstractions

Assessing yourself

- Worksheets

- https://drive.google.com/drive/folders/1TbCNnFREXuA7rkzhwWb7aMtpA2kytPum?usp=sharing

Iterating & Improving

- Debug yourself! Collect data on what your specific issues are and target them
  - Examples:
    - Forgetting to talk about complexity early on
    - Significant logic errors in coding
    - Blanking out, where & why?

- Have some core principles to remember and orient. Some of mine:
  - Senior engineering mindset : Data drives decisions. Gather data by conversations
  - “It is my responsibility you understand exactly what is going on”
  - Hyper aware of my tools & time
  - Patience and Trust in the early stages
  - Anxiety occurs when you don’t have a clear next step
  - “Small wins”. “Series of small wins is a big win”

Questions for me

- nirmalthacker@gmail.com

FastFunc