CRACKING THE CODING SKILLS

Created By Gayle Laakmann McDowell

Best Conceivable Runtime (BCR)

BCR is the runtime you *know* you can't beat. For example, if asked to compute the intersection of two sets, you know you can't beat O(|A|+|B|).

5 Approaches

- **BUD:** Look for bottlenecks, unnecessary work, duplicated work.
- **DIY:** Do It Yourself
- **Simplify & Generalize:** Solve a simpler version.
- **Base Case & Build:** Solve for the base cases then build from there.
- **Data Structure Brainstorm:** Try various data structures.

1 Listen

Pay very close attention to any info in the problem description. You probably need it all for an optimal algorithm.

2 Example

Most examples are too small or are special cases. **Debug your example.** Is there any way it's a special case? Is it big enough?

3 Brute Force

Get a brute-force solution as soon as possible. Don't worry about developing an efficient algorithm yet. State a naive algorithm and its runtime, then optimize from there. Don't code yet though!

4 Optimize

Walk through your brute force with **BUD optimization** or try some of these ideas:

- Look for any unused info. You usually need all the information in a problem.
- Solve it manually on an example, then reverse engineer your thought process. How did you solve it?
- Solve it "incorrectly" and then think about why the algorithm fails. Can you fix those issues?
- Make a time vs. space tradeoff. Hash tables are especially useful!

BUD Optimization

Bottlenecks

Unnecessary Work

Duplicated Work

5 Walk Through

Now that you have an optimal solution, **walk through your approach in detail.** Make sure you understand each detail before you start coding.

6 Implement

Your goal is to **write beautiful code.** Modularize your code from the beginning and refactor to clean up anything that isn't beautiful.

7 Test

Test in this order:

1. Conceptual test. Walk through your code like you would for a detailed code review.
2. Unusual or non-standard code.
3. Hot spots, like arithmetic and null nodes.
4. Small test cases. It's much faster than a big test case and just as effective.
5. Special cases and edge cases.

And when you find bugs, **fix them carefully!**

What You Need To Know

1 **Data Structures:** Hash Tables, Linked Lists, Stacks, Queues, Trees, Tries, Graphs, Vectors, Heaps.

2 **Algorithms:** Quick Sort, Merge Sort, Binary Search, Breadth-First Search, Depth-First Search.

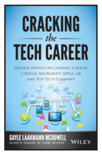
3 **Concepts:** Big-O Time, Big-O Space, Recursion & Memoization, Probability, Bit Manipulation.

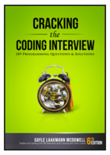
Exercises:

- Implement data structures & algorithms from scratch.
- Prove to yourself the runtime of the major algorithms.

Do not...

- **Do not** ignore information given. Info is there for a reason.
- **Do not** try to solve problems in your head. Use an example!
- **Do not** push through code when confused. Stop and think!
- **Do not** dive into code without interviewer "sign off."





Books by Gayle

CRACKING THE SOFT SKILLS

Created By Gayle Laakmann McDowell

Must Knows

- "So, tell me a bit about yourself…"
- "Why do you want to work here?"
- "Why should we hire you?"
- "Why are you leaving your current job?"
- "Where do you see yourself in 5 years?"
- "What do you do outside of work?"
- "What are your strengths and weaknesses?"

Your Two-Minute Pitch

"I am a ____ at ______."

"In college, I studied ___ at ____."

"Then I worked for … where I …."

"Then I worked for … where I …."

"In my current job, I've accomplished…."

"Also, outside of work, I …."

Preparing for Behavioral Questions

Create Resume Grid → Pick Five Key Stories → Diagram Your Stories

Themes	Job 1	Job 2
Leadership & Influence	*story*	
Mistakes & Failures		*story*
Challenges	*story*	
Teamwork	*story*	*story*
Successes		*story*

Coder? *Add: Bugs, Architecture, Optimization, Scaling.*

Stories	Story 1	Story 2
THEME(S)	*e.g., Leadership, Challenge, …*	
Nugget	*"Sure, let me tell you about when I …"*	
Situation	*Only the basics needed*	
Action(s)	*Expand here!*	
Result	*Prove it!*	
THE POINT	*What does it say about you?*	

Structured Answers

N — **Nugget.** "Sure, let me tell you about the time that I …" This focuses you and your interviewer on what you're about to say.

S — **Situation.** Explain just basics. The interviewer only needs enough details to understand what you did. Most people spend too much time here.

A — **Action(s).** Detail what actions you took. "First, I …. Then, I …. And finally, I …." This is where you should spend most of your time. Speak in bullets!

R — **Result.** Succinctly explain the result of your efforts were. Prove that the impact was good with numbers or a clear success metric.

Check Your Stories

- Are they substantial?
- Are they understandable?
- Have you explained *why* you did it this way?
- What do they say about you?
- Are they really about you (not your team)?
- Have you covered all the themes?
- Can you answer, "What would you do differently?"?

Questions For Your Interviewer

- Things you want to know.
- Things that show passion/interest.
- Things that show skills.

CRACKING THE PM SKILLS

Created By Gayle Laakmann McDowell

Estimation Questions

How many golf balls can fit in a 747? How many keyboards are sold in a year?

1. Clarify.
2. Think about what you know.
3. Make an equation.
4. Consider significant edge cases.
5. Solve components.
6. Sanity Check.

- Use nice, round numbers.
- Don't get too detailed. You're only going for a ballpark estimate.

Product Questions

Design a …

1. Ask questions.
2. Provide a structure.
3. Identify users.
4. Describe use cases & goals.
   - If altering existing product, how well does it meet user needs?
5. Design the product.

Improve …

1. Identify product goal.
2. Describe product issues.
   - Put users first!
3. Explore possible solutions.
4. Describe implementations of solutions.
5. Explain how you would validate implementations are successful.

Favorite product and why

1. What problems does it solve?
2. How does the product accomplish these goals? Why do you love it?
3. How does it compare to the alternatives?
4. How would you improve it?

Preparation

- Analyze your favorite physical product, website, and mobile app.
- Understand metrics: users, traffic, referral, engagement, retention, revenue, costs, etc.
- Analyze company's product: users, goals, strengths, challenges, competitors, tradeoffs.

FOCUS ON THE USER!

Case Questions

Marketing

- First analyze the company, competitors, customers, and landscape.
- Then design your marketing plan to fit goals.

Launching a Product

1. Discuss vision for product.
2. Determine goals of launch.
3. Design overview of launch.
4. Plan pre, during, & post-launch.

Strategy

- Macro: What is the product's strategy?
- Micro: How does the product align with company strategy?
- Think: mission, goals, strengths, weaknesses.

Problem Solving

1. Isolate the problem.
2. Diagnose the cause.
3. Solve the problem.
4. Consider tradeoffs.

Frameworks: Customer Decision Making Process, Marketing Mix (4 P's), SWOT Analysis, Situational Analysis (5 C's), Porter's Five Forces. Rarely directly useful, but good inspiration to create your own framework!

Pricing and Profitability

- Calculate with: cost-plus pricing, value pricing, competitive pricing, experimental pricing.
- Structures: ad-supported, freemium, tied, a la carte, subscription, free trials, razor blade model.

Brainstorming: How many things can you do with a paperclip?

- Don't worry about a stupid idea. Be creative!
- Strengths and Key Assets: A paperclip is thin, bendy, metal, light, pokey, etc.
- One vs. Many: What can you do with one paperclip? What if you had many?
- As-Is vs. With Modifications: What can you do with a paperclip as-is? What if you can modify it (melt it down, etc.)?

D **Drive.** Don't make your interviewer ask a bunch of follow-ups. Take charge. Drive, don't ride.

I **Instincts:** Show good instincts. Your interviewer is testing how you'd perform without all the research.

F **Framework:** Structure is key. Come up with your own framework to tackle each problem.

It's always about …

- **Structure:** Demonstrate that you can break down a problem and discuss it in an organization fashion.
- **User-Focused Thinking:** Put yourself in the user's shoes and think about what they want.
- **Problem-Solving Ability:** Companies want smart people.