CRACKING THE CODING SKILLS

Created By Gayle Laakmann McDowell

1 Listen

Pay very close attention to any information in the problem description. If it's given, you need it.

2 Example

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

3 Brute Force

State a brute force solution as soon as possible. Think about what the **best conceivable runtime** (BCR) looks like. Your final solution will be between your current one and the BCR.

4 Optimize

Walk through your brute force with **BUD optimization**, or try the **four algorithm approaches** (yellow box). Still stuck? Try these things:

- Look for any unused info.
- Use a fresh example.
- Solve it "incorrectly."
- Make time vs. space tradeoff.
- Precompute or do upfront work.
- Try a hash table or another data structure.

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. Does it do the right thing?
2. Weird looking code.
3. Hot spots.
4. Small test cases. Your example from #2 makes a bad test case.
5. Special cases.

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

BUD Optimization

Bottlenecks

Unnecessary Work

Duplicated Work

Best Conceivable Runtime (BCR)

BCR is a lower-bound on the runtime of a problem's solution. For example, the BCR of computing the intersection of two sets (A and B) is O(|A|+|B|). You know you can't beat that.

4 Approaches

- **Pattern Matching:** What problems is this similar to?
- **Simplify & Generalize:** Tweak and solve simpler problem.
- **Base Case & Build:** Does it sound recursive-ish?
- **Data Structure Brainstorm:** Try various data structures.

What You Need To Know

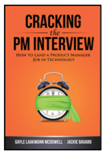
1. **Data Structures:** Hash Tables, Linked Lists, Stacks, Queues, Trees, Tries, Graphs, Vectors, Heaps.
2. **Algorithms:** Quick Sort, Merge Sort, Binary Search, Breadth-First Search, Depth-First Search.
3. **Concepts:** Big-O Time, Big-O Space, Recursion & Memoization, Probability, Bit Manipulation.

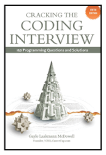
Exercises:

- 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