Recursion Workbook for CS 2

Jeremy Evert

November 3, 2025

Contents

1

Welcome to Recursion

The Big Picture

Welcome to Chapter 14 — where loops learn to dream bigger.

Up to now, you've been mastering the building blocks of programming:

- In **Chapters 1–4**, you learned to build small decisions with logic and branching.
- In **Chapters 5–6**, you harnessed repetition through loops and functions.
- **Chapters 7–12** introduced ways to store, structure, and reuse data and code: strings, lists, dictionaries, modules, and files.
- In **Chapter 13**, you explored inheritance — the first taste of elegant self-reference in object-oriented programming.

Now comes recursion — the art of a function that calls itself. It's not just another way to repeat something; it's a deeper way to think.

Why Recursion Matters

Recursion is the moment when programming starts to feel like storytelling:

> "To solve this problem, I'll solve a smaller version of the same problem, until it becomes so simple it solves itself."

This chapter is where abstraction and problem-solving meet. Recursion helps you:

1. Break large problems into smaller, self-similar ones.
2. Write cleaner code for structures that naturally branch — like trees, directories, or nested data.
3. Understand the mathematical elegance behind algorithms like Fibonacci, quicksort, and binary search.

How It Fits the Course Flow

Recursion bridges **loops and algorithms**. Think of it as a new dimension added to functions:

$$\text{iteration} \Rightarrow \text{recursion} \Rightarrow \text{algorithmic thinking.}$$

By the end of this unit, you'll be able to:

- Identify when recursion is a good fit (and when it's not).
- Trace recursive calls like a detective following a trail of function frames.
- Design your own recursive algorithms for search, sorting, and pattern exploration.

What's Ahead in Chapter 14

Here's how this section aligns with your ZyBooks topics:

14.1 Recursive Functions
Learn the structure of a recursive definition.

14.2 Recursive Algorithm: Search
Explore how recursion simplifies search logic.

14.3 Debugging Recursion
Learn to use print statements to trace your way through the stack.

14.4 Creating a Recursive Function
Practice building and testing your own.

14.5 Recursive Math Functions
Apply recursion to classic math problems.

14.6 Exploration of All Possibilities
See how recursion enables exhaustive search.

14.7–14.8 Labs
Build Fibonacci and permutation generators — your first recursive masterpieces.

Mindset for Success

When you first see recursion, your brain may shout:

"Wait — it's calling itself? But... how does it stop?"

That's normal. Everyone wrestles with the base case and the recursive step. Recursion feels like magic until you learn the trick — and then you realize you've been doing it all along: thinking, teaching, and even living recursively.

So take a breath, trust the process, and remember:

Every problem that feels too big... can be made smaller.

Welcome to recursion. Let's dive down the rabbit hole.

Notes

Use this space for your own discoveries and recursive experiments.