Math Foundations for Developers

From Zero to Calculus-Ready

Welcome! This curriculum is designed specifically for software developers who want to rebuild their math foundations from the ground up. Whether you're preparing for machine learning, computer graphics, algorithms, or just want to feel confident around mathematical concepts, you're in the right place.

Who This Is For

You're a software developer who:

- Thinks in code: variables, functions, data structures
- Has around grade 8 level math (or less, and that's totally fine)
- Wants to understand *why*, not just memorize formulas
- Learns best with concrete examples and visual models
- Is tired of "just trust me" explanations

What You'll Learn

By the end of this curriculum, you'll:

- Understand numbers from first principles
- Be comfortable with algebra, functions, and graphs
- Actually understand logarithms (not just copy-paste them)
- Know what derivatives and integrals *mean* (without hardcore calculus)
- Feel confident reading technical papers and ML documentation
- See connections between math and code everywhere

How to Use This Curriculum

Linear Path (Recommended)

Read the topics in order. Each builds on previous concepts:

1. **00-numbers.md** - How numbers actually work
2. **01-algebra-foundations.md** - Variables and equations
3. **02-ratios-proportions-percentages.md** - Comparisons and scaling
4. **03-powers-roots-exponents.md** - Repeated multiplication
5. **04-logarithms.md** - The inverse of exponentials
6. **05-coordinate-geometry.md** - Points and graphs
7. **06-functions.md** - The core of everything
8. **07-linear-functions.md** - Straight lines and slopes
9. **08-trigonometry.md** - Angles, rotation, waves
10. **09-polynomials.md** - Curves and roots
11. **10-limits.md** - Gateway to calculus
12. **11-derivatives.md** - Rate of change
13. **12-integrals.md** - Accumulation and area

Study Tips

Go Slow: Don't rush. Each topic is designed to be digested in one sitting, but take breaks.

Do the Examples: The tiny practice sections aren't homework—they're checkpoints to ensure you're getting it.

Use the Analogies: Programming analogies aren't decoration. They're bridges between what you know (code) and what you're learning (math).

Draw Everything: Math becomes clearer when you visualize it. Sketch the diagrams yourself.

No Shame: If something doesn't click immediately, that's normal. Come back to it later with fresh eyes.

Philosophy

This curriculum follows three principles:

1. Intuition First, Formulas Second

You'll always understand *why* before you see *what*. Math isn't magic spells—it's tools that solve real problems.

2. Zero Assumptions

Every symbol gets explained. Every step gets justified. No "it's obvious" or "clearly."

3. Mental Models Over Memorization

You'll build internal representations that stick. When you forget the formula (and you will), you can rebuild it from the mental model.

What This Isn't

- **Not a textbook**: No theorem-proof-corollary structure
- **Not exam prep**: No problem sets with 50 similar questions
- **Not formal**: We care about understanding, not mathematical rigor
- **Not rushed**: Take your time. This isn't a race.

Programming Mindset Throughout

You'll see constant parallels to programming:

- Variables in math work like parameters in functions
- Equations are like solving for unknowns in constraints
- Functions in math are similar to pure functions in code
- Graphs are like data visualizations
- Derivatives are like rate of change in animations
- Integrals are like `reduce()` operations

What You Need

- A text editor or Markdown viewer (you're already here!)
- Pen and paper for sketching
- A calculator (or Python REPL, or browser console)
- Curiosity and patience

Getting Started

Open 00-numbers.md and begin your journey.

Remember: You don't need to be "good at math" to understand math. You just need to be willing to think slowly and carefully.

Let's rebuild your mathematical intuition from the ground up.

Questions, corrections, or feedback? This is a living document. Improve it as you go.