Opinion

AI and students

I use AI for most of my coding because I spent decades learning the foundations first. Students and beginners need a different approach: use AI to support the work without handing over the thinking.

Fabio Akita · ailair.akitaonrails.com/students/

> *Me, a senior with years of experience, I'll vibe code 100%. You, a student, a junior, MUST write as much as possible by hand, let the AI do the more mundane tasks in parallel, and you MUST review everything manually. You need tens of thousands of hours of deliberate practice and study to become someone who later knows what to ask the AI to do for you. Not before.*
>
> *Eu, sênior, anos de experiência, vou fazer 100% vibe codar. você, estudante, júnior, DEVE escrever o máximo possível na mão, deixar a IA fazer tarefas mais mundanas, em paralelo, e você DEVE revisar tudo manualmente. Você precisa de dezenas de milhares de horas de prática deliberada, estudo, pra se tornar alguém que, depois, vai saber o que mandar a IA fazer por você, não antes.*
>
> ***Me on X, September 24, 2026***

1. The multiplier

AI multiplies what you bring

The same tool gives a senior more good work, faster, and gives a beginner more of whatever the beginner has, mistakes included.



1. The multiplier

AI multiplies what you bring

- AI can multiply mistakes when the foundations are weak. With strong foundations, it can multiply useful work.
- Google's DORA 2025 report found the same in teams: "AI doesn't fix a team; it amplifies what's already there."
- If you're experienced and still resisting, stop. There's no merit in writing a Dockerfile or a Makefile by hand for the hundredth time. The chores go to AI; the foundations don't.
- People who understand operating systems, databases, networks, data structures and compilers use agents as an exoskeleton. People who don't use them as a crutch. An exoskeleton amplifies strength; a crutch only hides weakness.
- And someone has to check the output. In Veracode's 2025 test of over 100 models, 45% of the code samples failed security tests and introduced OWASP Top 10 vulnerabilities.

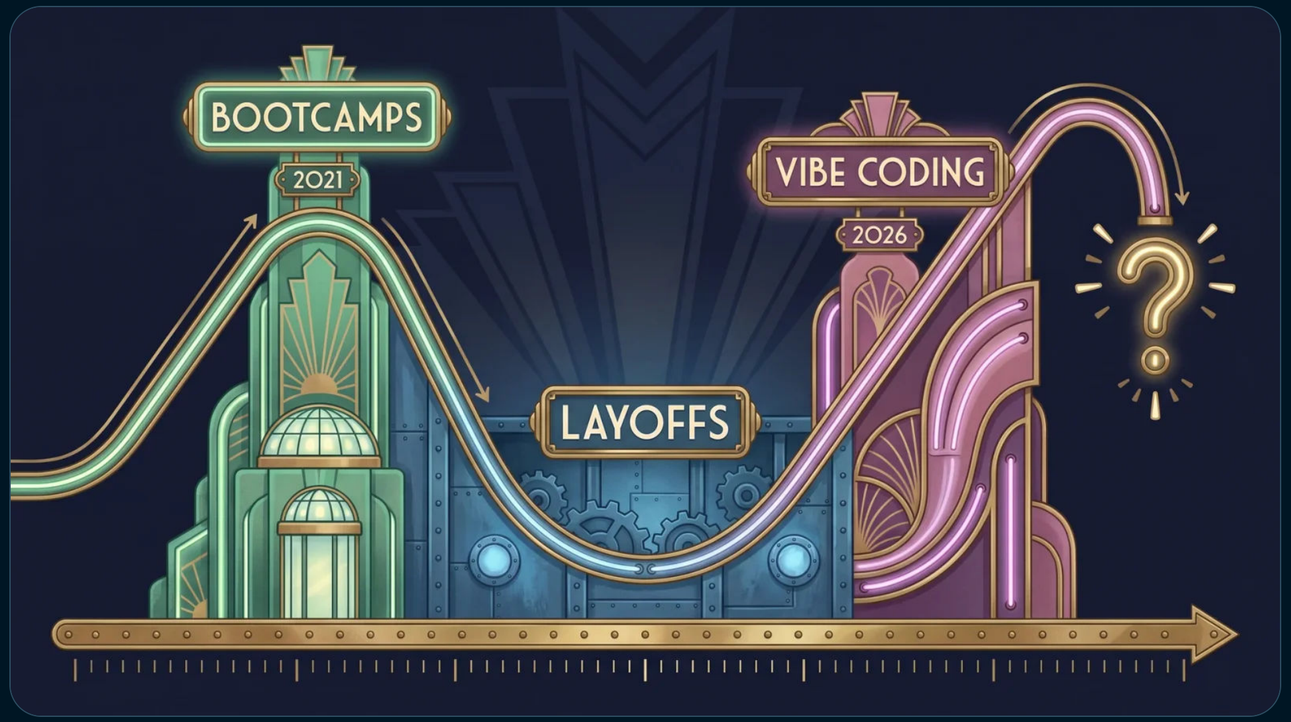
1. The multiplier

> *A well-prepared programmer won't be replaced. I said it clearly: I never will be.*
>
> *Programador bem preparado não vai ser substituído. Disse claramente: "EU" nunca vou ser.*
>
> ***Me on X, September 28, 2026***

2. The last bubble

We've seen this shortcut before

Before AI, the promise was a bootcamp: two months and you'd earn a Google salary. It ended in closures and layoffs.



2. The last bubble

We've seen this shortcut before

- Bootcamp graduates grew about eightfold between 2013 and 2018. Lambda School advertised placement rates "as high as 86 percent"; the Consumer Financial Protection Bureau found its internal numbers closer to 50%, sometimes as low as 30%.
- I started Akitando in 2018 for exactly this: to prepare real programmers to survive the bubble popping. In 2022 I was telling people to cancel the courses and stop believing influencers.
- Now the same promise is back with a new face: you don't need to learn to code, the AI does it for you. Nobody becomes an engineer in a one-month bootcamp, and nobody becomes one by prompting either.
- Then the cheap money ran out. Bootcamps closed in 2023 and 2024, 2U announced in late 2024 it was leaving the business, and the tech industry cut more than 150,000 jobs that year alone.
- What those courses taught, forms, CRUD, reports and copies of the same UI, is exactly what LLMs replicate best. Most of those jobs were already cut after 2022.

3. The evidence

Learning on autopilot doesn't stick

The studies so far point the same way: when the AI does the thinking, you don't learn.

17 pts

fewer quiz points for mostly junior engineers who learned a new library with AI

17%

lower exam scores for students who practiced with plain GPT-4, once it was gone

19%

slower: experienced developers with AI, who still believed they were faster

3. The evidence

Learning on autopilot doesn't stick

- Anthropic's own study, January 2026: mostly junior engineers learning a new library (Trio) with AI scored 50% on the quiz, against 67% for those who coded by hand, with no significant time saved. Those who used AI to explain concepts kept most of the learning; those who handed off the code scored among the lowest.
- METR's 2025 trial: experienced open-source developers took 19% longer with AI, and afterwards still believed it had made them 20% faster.
- In a PNAS study with about 1,000 high school students, plain GPT-4 raised practice scores by 48% and then cut exam scores by 17% once it was taken away. A tutor built to make them think largely avoided the loss.
- An MIT preprint, not yet peer reviewed, found that people who wrote essays with ChatGPT struggled to quote their own work.

4. The market

The entry-level path is narrower

Companies still hire people who can review and decide. They hire far fewer beginners to type.

65%

less new-grad hiring at the big tech companies than in 2019

4.5%

of software development job postings are entry-level (Q1 2026)

19%

fewer 22 to 25 year-olds than expected in the most AI-exposed jobs

The entry-level path is narrower

- SignalFire, 2026: new-grad hiring is down about 65% at the big tech companies and 76% at early-stage startups compared with 2019, while engineering hiring at the big tech companies is down only 11%.
- Stanford's "Canaries" study: employment for 22 to 25 year-olds in the most AI-exposed jobs, software developers included, is 19% below where it would be had it kept pace with less-exposed jobs, mostly through fewer hires. The authors are careful to call it descriptive rather than causal.
- Indeed: entry-level roles were 4.5% of software postings in early 2026. Postings grew almost 15% after Claude Code launched in February 2025, and 71% of that growth was senior roles.
- Recent computer science graduates have one of the highest unemployment rates of any major (7.0%) but one of the lowest underemployment rates (19%, against 39% overall): when they're hired, it's usually into a job that needs the degree. The typical entry path is still a bachelor's degree.

5. The gap

AI can't guess what you need

It does what you ask, and it doesn't do what you didn't ask. Knowing what to ask is the job.



5. The gap

AI can't guess what you need

- A coding agent needs you to explain the goal, the context, the constraints, the priorities, what must not break, which paths are acceptable and how success is measured. Many programmers never learned to say that.
- Even for a tiny app, anything built by vibe coding that goes to production must be reviewed by an experienced programmer. That isn't optional.
- My way of talking to an agent has four parts: what I want, how, what I don't want, and how we check it. You can't fill those in without knowing the domain.

6. The language myth

Tests matter more than the language

In October 2026 I gave 4 models the same hard task in 7 languages. No language came out ahead.

112

attempts: 7 languages, 4 models, 2 conditions, 2 runs each

24

correct results from the two weaker models when I gave them the tests

16

correct results from the same models without the tests

6. The language myth

Tests matter more than the language

- The two frontier models got every eligible attempt right, in every language.
- Learn to write tests and turn every bug into one. That skill works in any language, with any model.
- Every failure broke a rule of the problem itself. None was a type error, so no compiler would have caught them.

7. The question

If AI can do all the programming, why learn it?

Because it can't do all of it. It reproduces what it has seen, and what it hasn't seen is exactly where the next jobs are.



7. The question

If AI can do all the programming, why learn it?

- Models learned from public code. Most of the world's code is private, 81.5% of GitHub contributions alone, and a problem nobody solved before isn't in any training set.
- Legacy is the same story. There's no big open source COBOL community: the code that runs banks, insurers and governments is private, and the models have barely seen it.
- In my view, that's the opportunity. The fewer people willing to learn the foundations the hard way, the more valuable the ones who do. They'll be the generation that closes the gaps the models can't close on their own.
- Where something similar was published, AI replicates it fast. Where it wasn't, it spins in long, expensive loops, and someone has to take over: define what "right" means, build the oracle, write tests with objective targets, find the reference nobody published, and notice when the agent starts going in circles.
- Every one of those jobs needs someone who understands the problem from the bottom: how computers, operating systems, compilers, networks and databases really work. That's the hard knowledge most people skip.

7. The question

> *The skill the machine needs most from you is the one it can least learn from its training.*
>
> *A habilidade que a máquina mais precisa de você é a que ela menos consegue aprender com o treino dela.*

8. A real example

What "the AI did it" actually took

ai-memory, my open source memory system for coding agents, is mostly written by AI. Everything that keeps it correct came from my review and direction, week after week.



8. A real example

What "the AI did it" actually took

4,743
words of rules for the agents in AGENTS.md

16
invariants that must never break

22
enforced security boundaries, each mapped to a guard and adversarial tests

3,000+
unit tests

600+
integration tests

- About 65 explicit directives, 28 curated memory notes that exist because of specific mistakes, and more than 60 documents with research, post-mortems and architecture decisions.
- Each layer marks a place where the AI alone didn't get it right and I had to know how to fix it.
- None of it showed up on day one. It's weeks of watching the AI do the bulk of the work every day and refining it, and the refining depends on experience.

8. A real example

> *None of this is automatic, made by the AI. All of it is where the AI alone can't do it and I need to know how to adjust. No amateur can do this.*
>
> *Nada disso é automático, feito pela IA. Tudo isso é onde a IA sozinha não consegue fazer e eu preciso saber com ajustar. Nenhum amador consegue fazer isso.*
>
> **Me on X, September 25, 2026**

9. What to do

If you're a student or a junior

Use AI the way a senior uses an intern: for the chores. The learning has to go through your own hands.



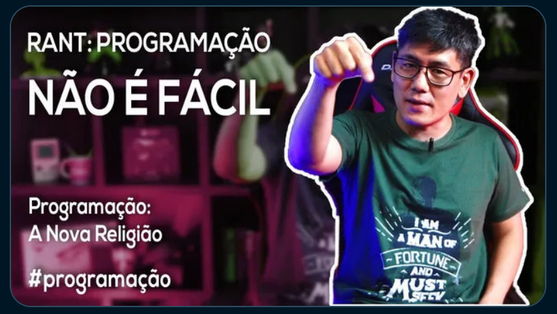
9. What to do

If you're a student or a junior

- Write as much as you can by hand. Break it, debug it, fix it. Getting stuck is part of it: Anthropic's researchers write that cognitive effort, "even getting painfully stuck", is likely important for mastery.
- Review everything it produces, line by line. If you can't explain a change, you're not ready to accept it.
- A senior is someone who has been a junior in every topic they master.
- Let AI handle the mundane tasks in parallel, and ask it to explain concepts, not to write the code you're supposed to learn from.
- Count the hours. From zero to a first junior job is at least about 1,400 hours of study and practice; in my view, mastery takes tens of thousands. Ericsson's best violinists had over 10,000 by age 20, and later research shows practice isn't the whole story, but nothing replaces it.

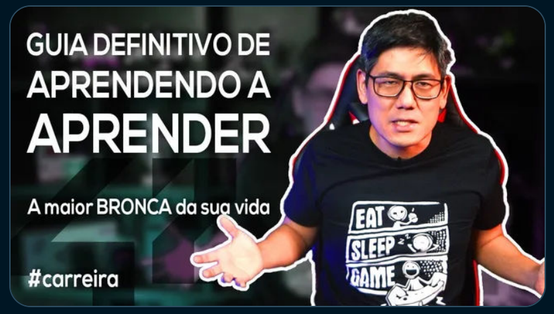
◈ Where to start

The Akitando archive



RANT: Programação NÃO É Fácil

Feb 2020 · 32:22



Guia DEFINITIVO de Aprendendo a Aprender | A maior BRONCA d...

Apr 2020 · 1:04:37



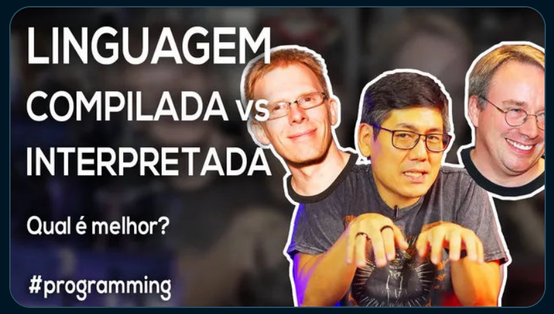
O Guia +Hardcore de Introdução à COMPUTAÇÃO

Jun 2020 · 1:18:28



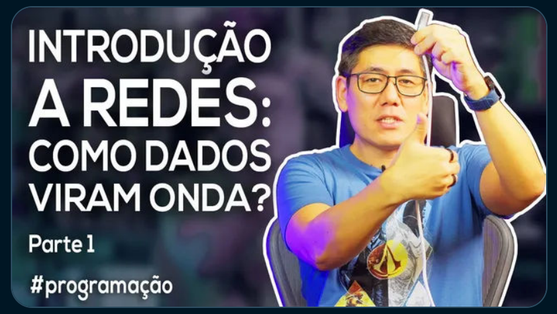
Hello World Como Você Nunca Viu! | Entendendo C

Mar 2021 · 1:09:22



Linguagem Compilada vs Interpretada | Qual é melhor?

Apr 2022 · 1:11:44



Introdução a Redes: Como Dados viram Ondas? | Parte 1

Jul 2022 · 37:59

The takeaways

1

AI multiplies what you already know. With little to multiply, you get very little.

2

Shortcuts sold as careers already had one crash. The new one is the same promise.

3

Learning on autopilot doesn't stick. The studies so far agree.

4

The work that remains valuable needs people who can judge the output, while entry-level hiring is shrinking.

5

AI can't guess your goal, your limits or your trade-offs. Knowing them is the job.

6

In my benchmark, giving the model tests changed the results. Pick the language that fits the problem.

7

AI replicates what's public. The gaps it leaves, in private and new work, are the job, and the hard foundations are how you fill them.

8

Write by hand, let AI do the chores, review everything, and put in the hours.

> *The only thing that matters is objective reality.*
>
> *A única coisa que importa é realidade objetiva.*

Read more: my posts and the sources

AI multiplies what you bring

- From zero to post-production in one week: using AI on real projects
- VS Code is the new punch card
- Stop making excuses and ship more
- Google DORA 2025: AI amplifies what's there
- Veracode: security flaws in AI-generated code

We've seen this shortcut before

- Rant: did AI kill programmers?
- CFPB: Lambda School's placement claims
- Course Report: the bootcamp market, 2013 to 2018
- Inside Higher Ed: the bootcamp closures
- Higher Ed Dive: 2U exits bootcamps
- TechCrunch: the 2024 tech layoffs
- Akitando #132: the startup bubble burst (2022, Portuguese)

Learning on autopilot doesn't stick

- New LLM Benchmark v4, Part 1: the process
- VS Code is the new punch card
- How DriveClub and shadPS4 almost defeated AI and me: how to learn
- Anthropic: how AI assistance affects coding skills
- Bastani et al., PNAS: generative AI can harm learning
- METR: experienced developers 19% slower
- MIT Media Lab: Your Brain on ChatGPT (preprint)

The entry-level path is narrower

- Rant: did AI kill programmers?
- SignalFire: State of Tech Talent 2026
- Indeed: the labor market tilts toward seniority
- Indeed: software postings since Claude Code (July 2026)
- Stanford Digital Economy Lab: the AI gap for young workers (August 2026)
- New York Fed: the labor market for recent graduates
- BLS: software developers

Read more: my posts and the sources

AI can't guess what you need

- Why LLMs will fail at your company
- Why LLMs aren't giving you the result you expect
- Vibe code: I built a little app 100% with GLM 4.7 (TV Clipboard)

Tests matter more than the language

- Are There Better Languages for Programming with LLMs?
- The benchmark: code, data and method

If AI can do all the programming, why learn it?

- The limits of LLMs: good at reproducing what exists, expensive for what doesn't
- AI Reverse Engineering: Is Adobe Dead?
- VS Code is the new punch card
- GitHub Octoverse 2025: 81.5% private
- arXiv (2026): LLMs and COBOL, code that is rarely public
- ARC Prize: new tasks, 0% for pure LLMs

What "the AI did it" actually took

- ai-memory 2.0
- Software is never done: four projects and life after deploy

If you're a student or a junior

- How DriveClub and shadPS4 almost defeated AI and me: how to learn
- RANT: Did Akita cave to AI?
- Wrapping up my AI marathon: success or failure?
- Anthropic: "getting painfully stuck"
- Ericsson et al. (1993): deliberate practice
- Macnamara and Maitra (2019): the replication
- Akitando #72: programming is not easy (Portuguese)

Every point, with every link, on the web

ailair.akitaonrails.com/students/

Fabio Akita · akitaonrails.com · x.com/AkitaOnRails