

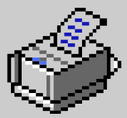




From COBOL to Cursor

Human language as the primary computer interface

https://brendan.fyi/cobol



03:31PM

nda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Agenda • Ager

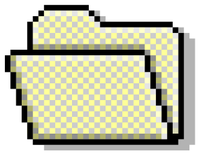
Agenda

Topics Covered

Amazing Grace

First Revolution

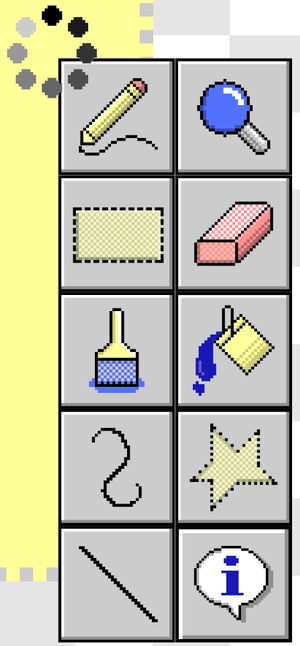
The way we've always done it



So what now?

Start

Amazing Grace



Back to Agenda Page

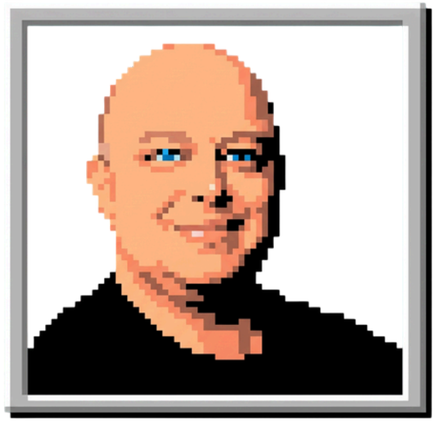
The most dangerous phrase in the English language is 'we've always done it that way.'

Admiral Grace Hopper
1906 - 1992

Back to Agenda Page

Every revolution looks like chaos to the incumbents

Grace Hopper spent 60 years proving that.
Let me show you.



Brendan O'Leary
Playing a developer on TV since 2008

Back to Agenda Page

Who was Grace Hopper?

7 Alarm Clocks

...at age 7 she took apart seven alarm clocks to figure out how they worked

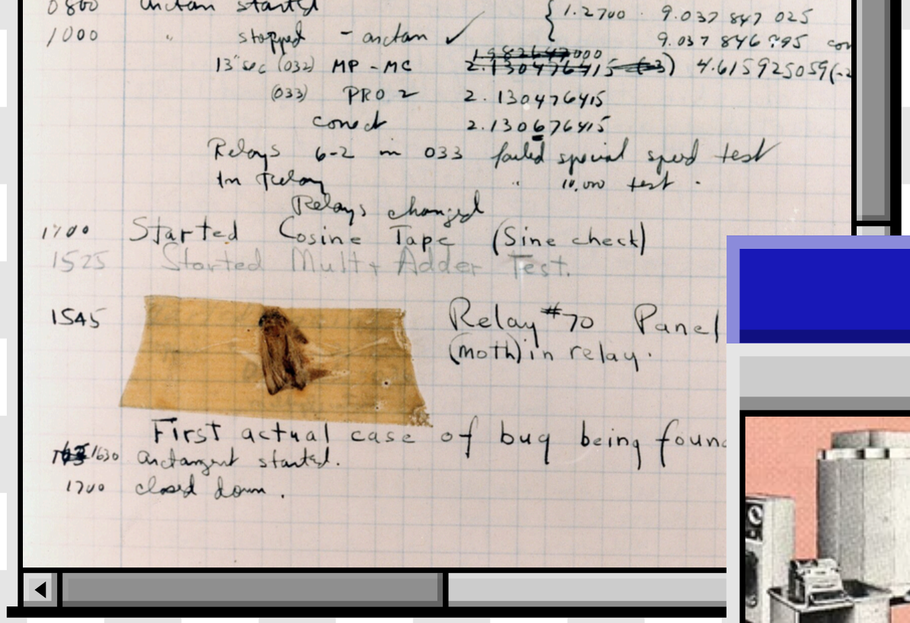
Yale, Vasser, Navy

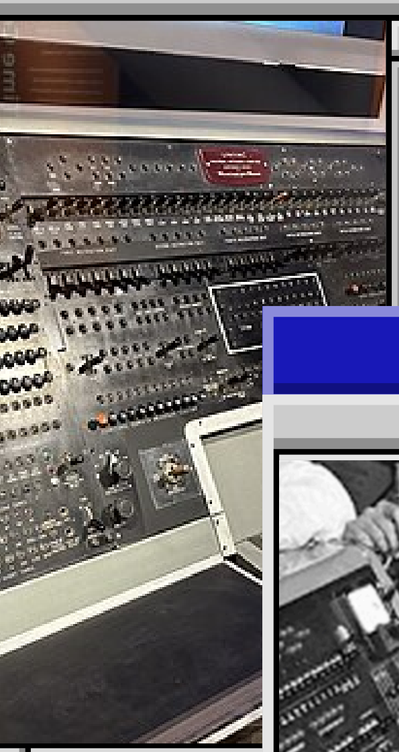
Underweight for the Navy, but got a special exemption

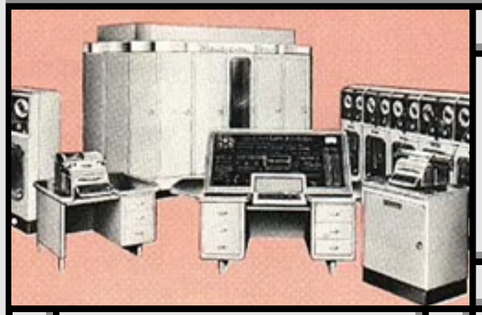
Admiral

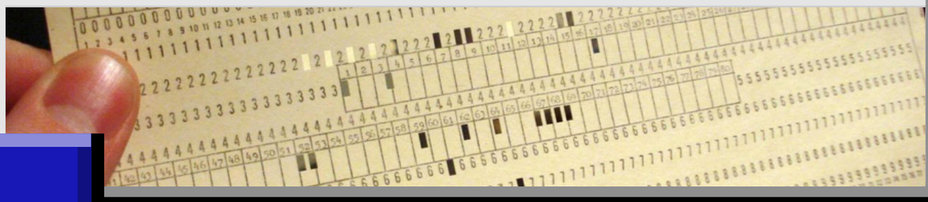
1966: Retired
1967: Recalled for a “6 month” assignment...until 1973

Back to Agenda Page







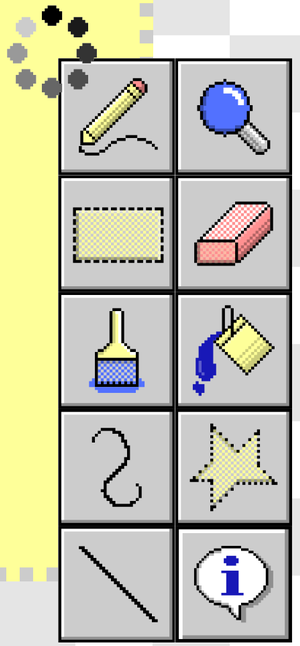


Back to Agenda Page

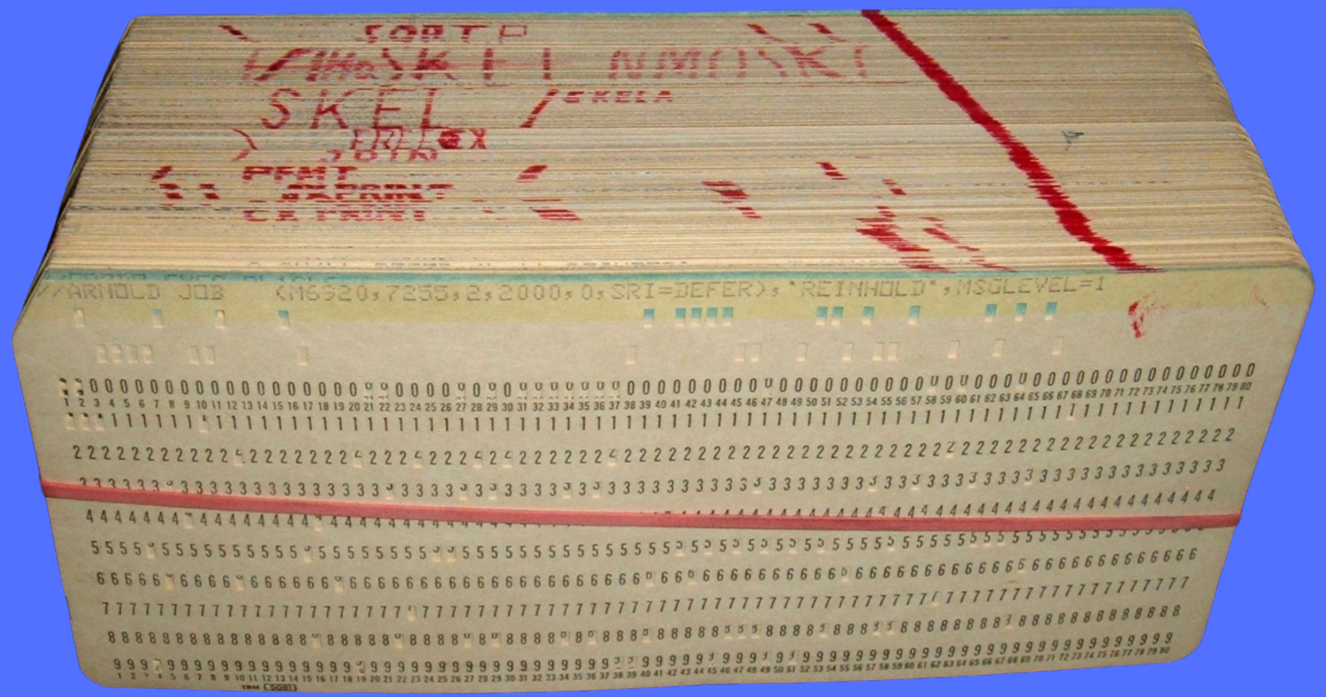
"There was no such thing as a programmer at that point. We had a code book for the machine and that was all."

Report to
CONFERENCE on DA
SYSTEMS LANGUAG
Including
INITIAL SPECI
for a COMMON B
ORIENTED LANGUAG
for Programm
Electronic Digital C
DEPARTMENT OF DEFENSE

The First Revolution



Back to Agenda Page



1952

State of the art computer programing

Back to Agenda Page

What I was after in beginning English language [programming] was to bring another whole group of people able to use the computer easily...
I kept calling for more user-friendly languages.

Admiral Grace Hopper
1906 - 1992

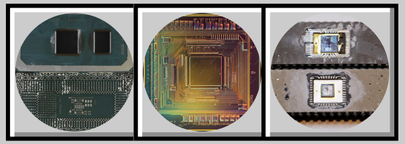
Back to Agenda Page

One machine
Punchcard, magnetic tape, etc.

One machine
Basically mnemonics for the binary code

Multi-architecture
More like English

Turtles all the way down



Machine Code

Assembly

COBOL

Next.JS
4GB node_modules

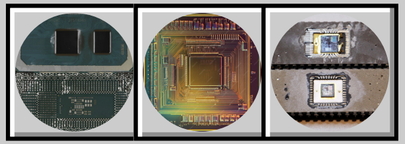
LLMs

Turtles all the way down

Machine Code

Machine Code

01000111



Assembly

Assembly

High-level languages

COBOL

LLMs

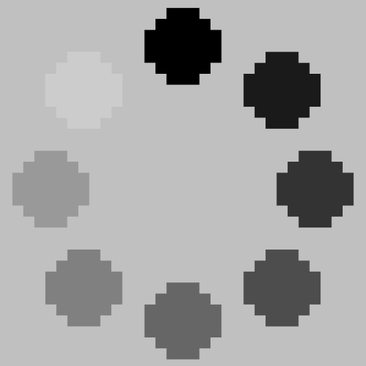
Next.JS
4GB node_modules

The way we've "always" done it

Back to Agenda Page

The way we've always done it

"But Grace, then anyone will be able to program!"



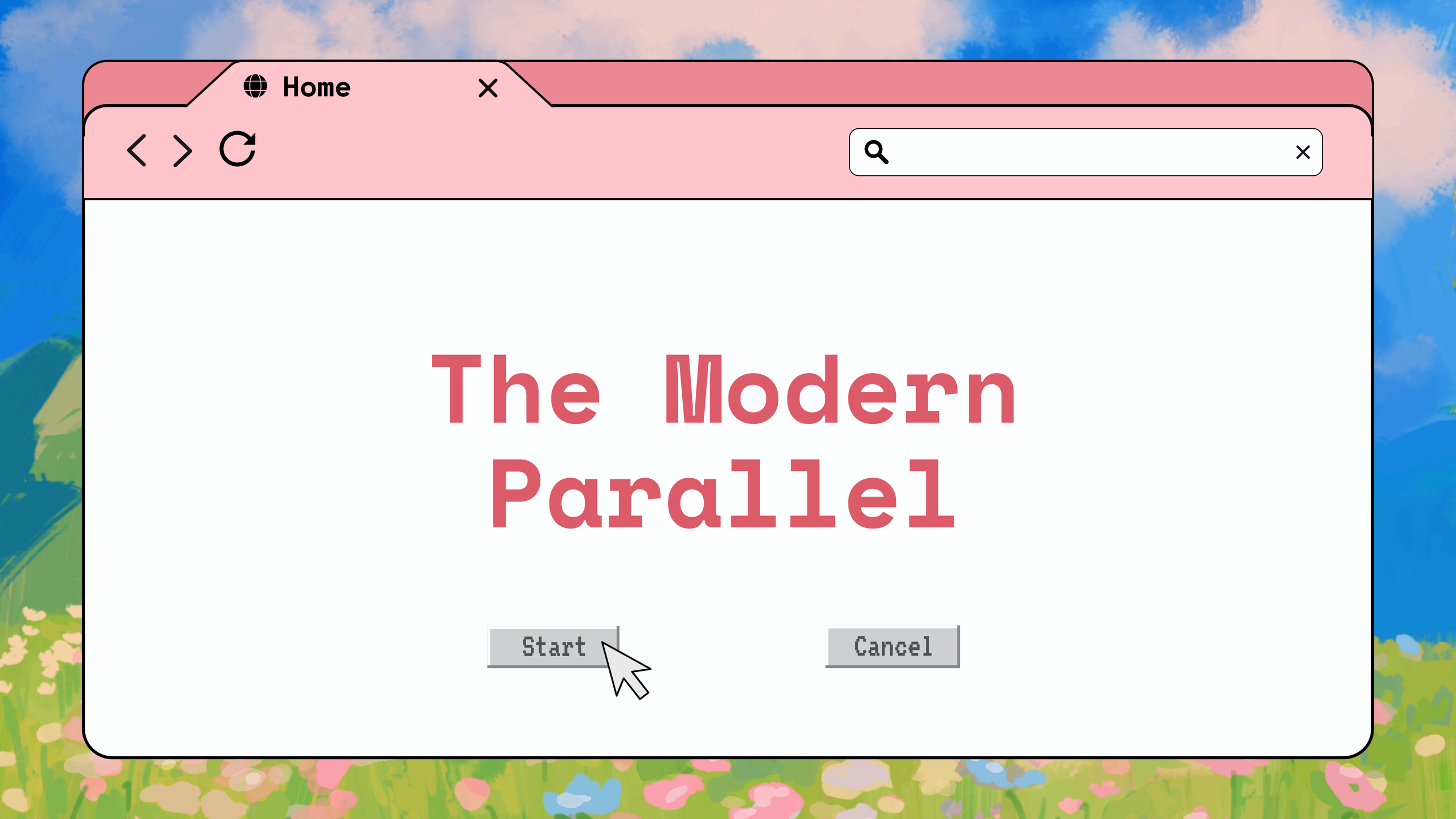
Exactly

"That's not real programming"

1950s	Punchcards → Assembly
1960s	Assembly → COBOL
1990s	COBOL → Python
2020s	Python → Prompts

Back to Agenda Page

~~The way we've always done it~~


Home
The Modern Parallel
Start
Cancel

Vibes

Vibes vs. Engineering

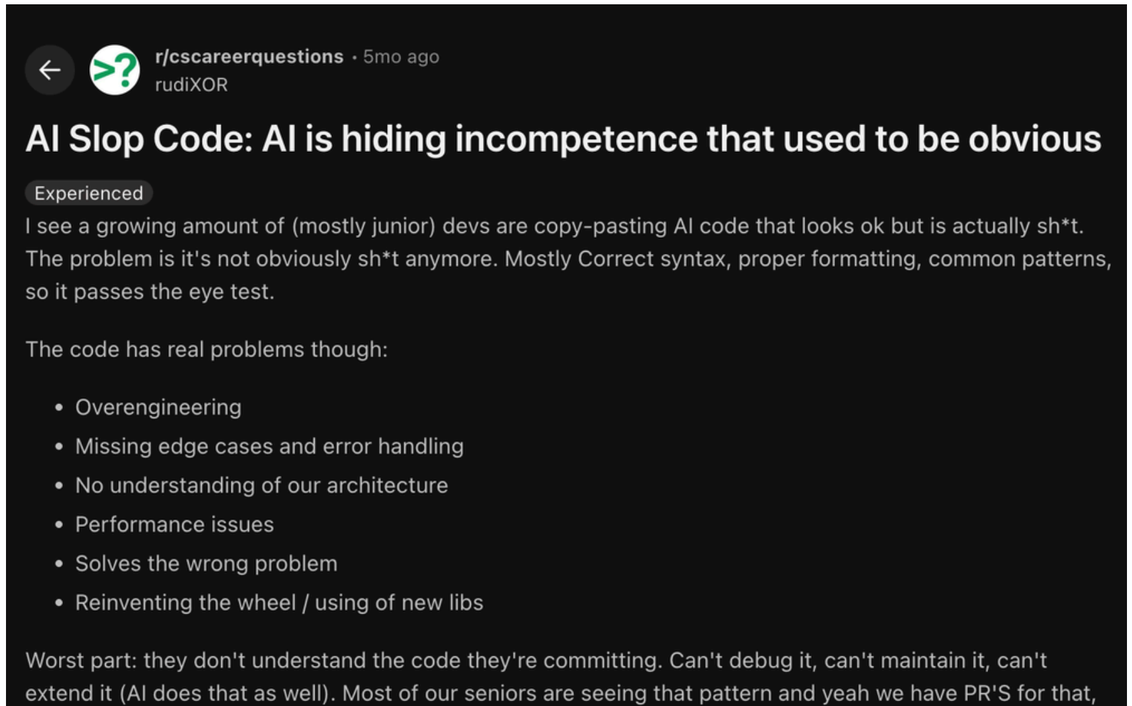
Yes

Ok

Andrej Karpathy
@karpathy

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

6:17 PM · Feb 2, 2025 · **6.8M** Views



r/cscareerquestions · 5mo ago
rudiXOR

AI Slop Code: AI is hiding incompetence that used to be obvious

Experienced

I see a growing amount of (mostly junior) devs are copy-pasting AI code that looks ok but is actually sh*t. The problem is it's not obviously sh*t anymore. Mostly Correct syntax, proper formatting, common patterns, so it passes the eye test.

The code has real problems though:

- Overengineering
- Missing edge cases and error handling
- No understanding of our architecture
- Performance issues
- Solves the wrong problem
- Reinventing the wheel / using of new libs

Worst part: they don't understand the code they're committing. Can't debug it, can't maintain it, can't extend it (AI does that as well). Most of our seniors are seeing that pattern and yeah we have PR'S for that,

It can be confidently wrong

Trust but verify: You don't trust junior devs' code without review. You don't trust Stack Overflow answers without testing. You don't trust your own code without tests.

AI shouldn't be held to a different standard—it should be held to the same standard.

Mistakes

It's making engineering a commodity

Every abstraction layer was supposed to reduce the need for programmers. COBOL was going to let business people program directly. Visual Basic was going to eliminate coders.

The result every time: more programmers, solving harder problems, because accessible tools expand what's possible.

Mistakes

You can't trust it for anything important

Do you read the assembly your compiler generates? Do you audit the JIT output of your runtime? At some point, you trust tools that transform your intent into execution.

The question is where to draw the line, and that line has always moved.

A COMPUTER

CAN NEVER BE HELD ACCOUNTABLE

THEREFORE A COMPUTER MUST NEVER

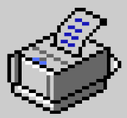
MAKE A MANAGEMENT DECISION



In 1980, Grace said: "I kept calling for more user-friendly languages."

Forty-five years later, the user-friendliest language is...

English.



Back to Agenda Page

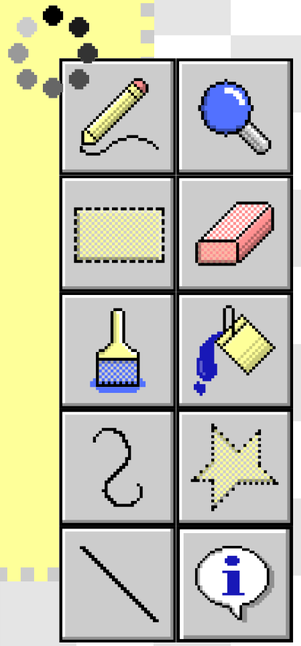
1983

“We’ve got the Model T. That’s where we are now”

Back to Agenda Page

So...
what now?

Back to Agenda Page



No “fix this bug” or “build it...no mistakes”

Patterns, examples, build good habbits

Building an intuition for when to stop, start, restart, push further


Expressing Intent
Software engineering has always been about
expressing intent in a structured way

No “fix this bug” or “build it...no mistakes”

Patterns, examples, build good habbits

Building an intuition for when to stop, start, restart, push further

“Ask” Mode

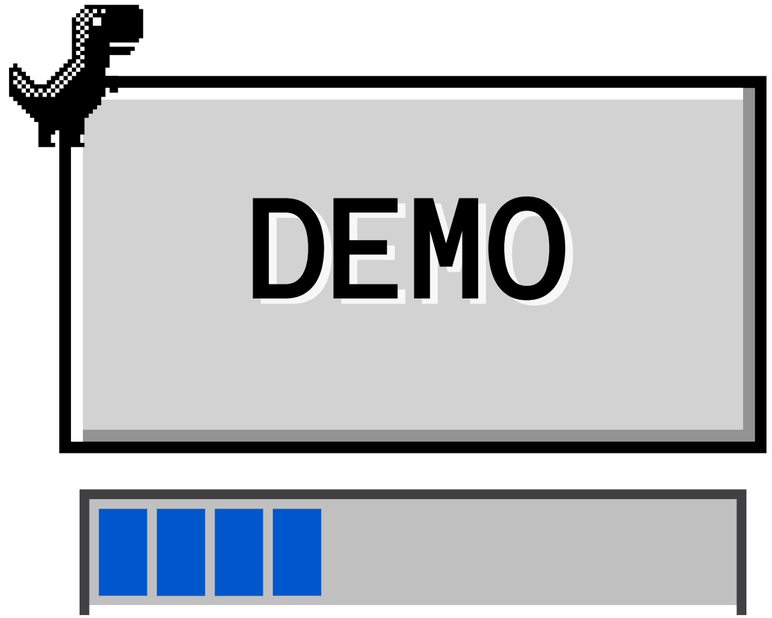
Research Plan Implement

- Understand the system
- Find the right structure
- Stay objective

Human-reviewed research results

“Ask” Mode

Research Plan Implement

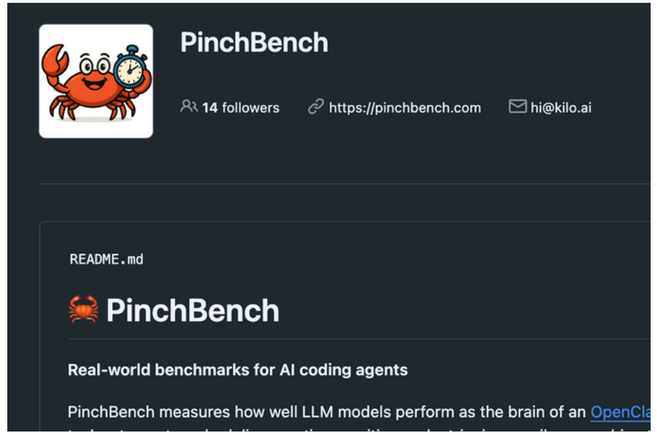


“Ask” Mode

Research Plan Implement

OPENCLAW

DEMO

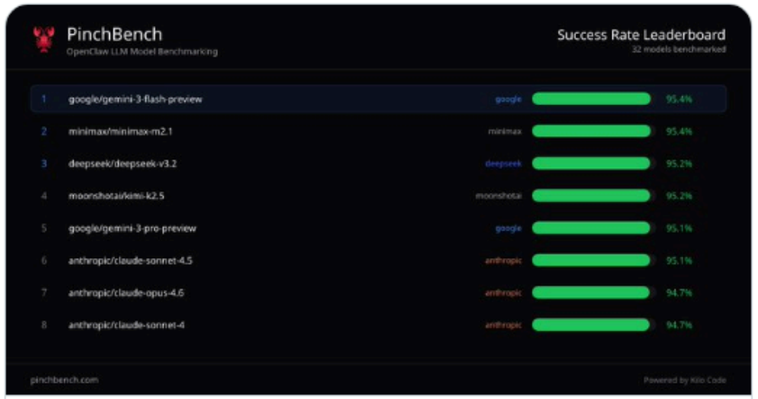


“Ask” Mode

Research Plan Implement

Peter Steinberger
@steipete · **Follow**

Interesting benchmark on which model is best for @openclaw

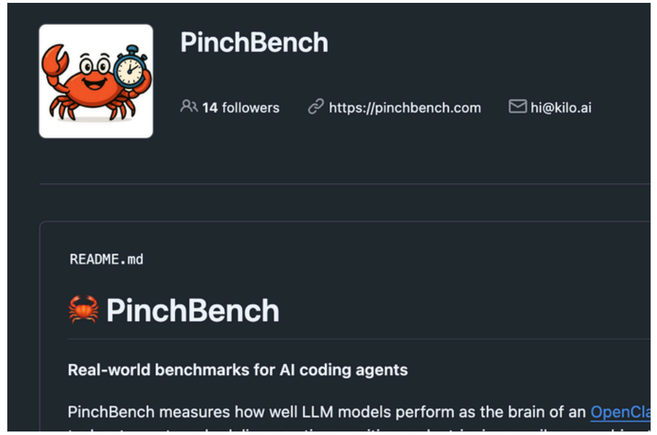


pinchbench.com
PinchBench - Success Rate Leaderboard
Benchmarking LLM models as AI agents across standardized coding tasks

3:58 PM · Mar 7, 2026

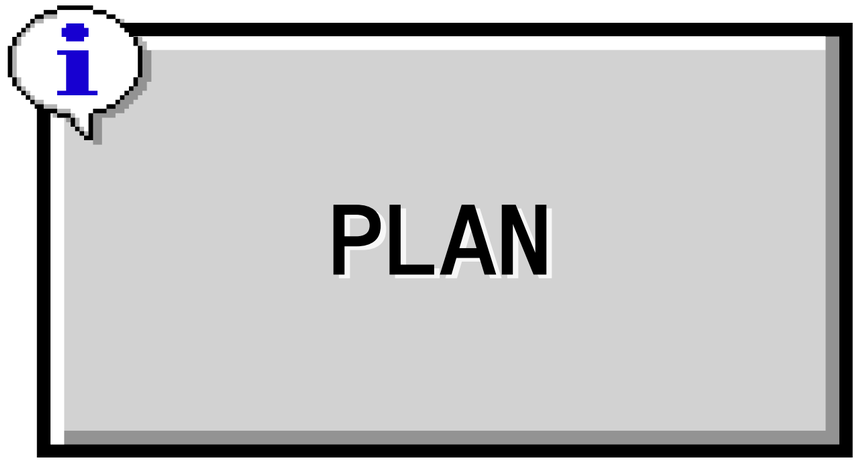
2.6K Reply Copy link

Read 326 replies



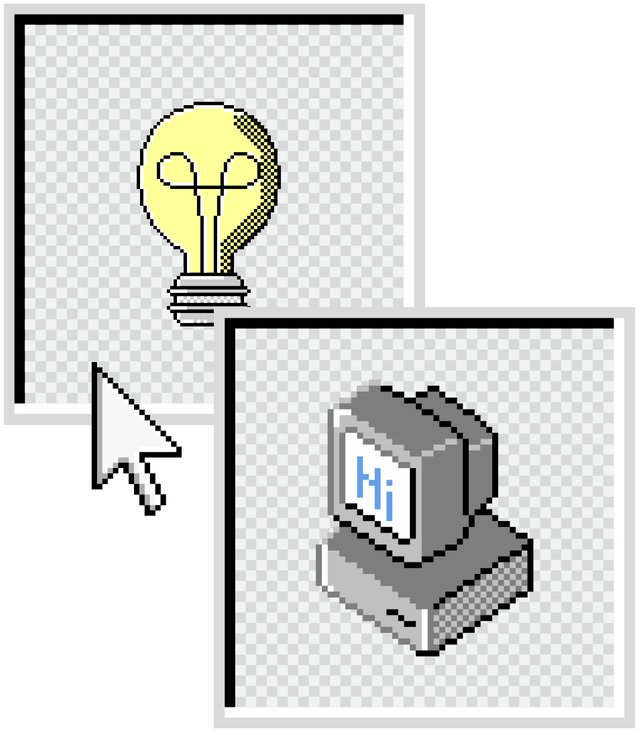
“Plan” Mode

Research **Plan** Implement



- Outline exact steps
- Include plans for testing
- Be explicit

Human-reviewed plan file

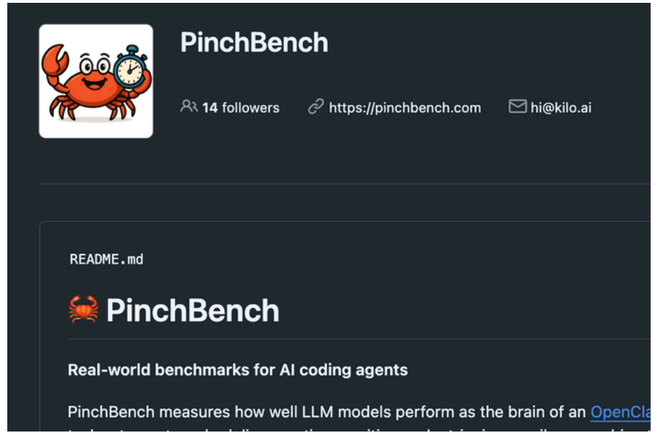


“Plan” Mode

Research **Plan** Implement

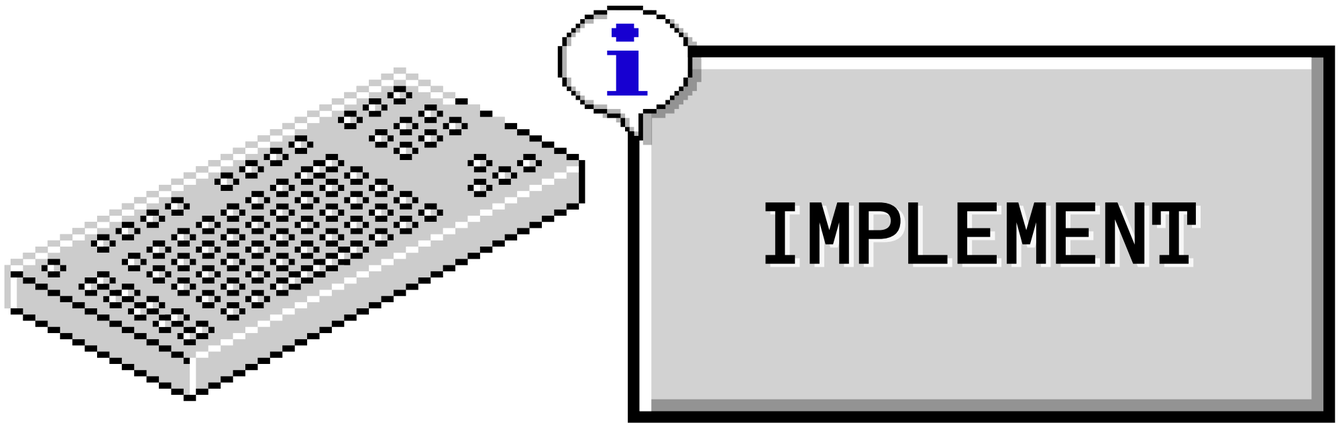
OPENCLAW

DEMO



“Code” Mode

Research Plan **Implement**



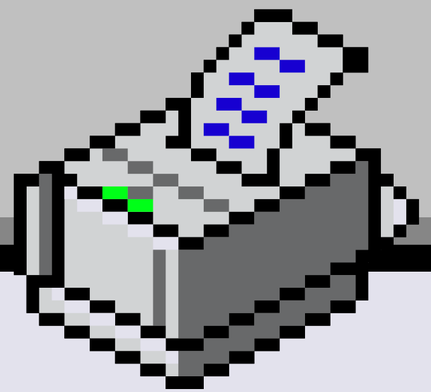
- Keeps context low
- Review each change
- Commit frequently

Human review at the **research** and **planning** stages is the highest leverage use of your time

FURTHER LISTENING

“AI cannot replace thinking. It can only amplify the thinking you have done – or the lack of thinking you have done.

– Dex Horthy



The old disciplines matter more with AI, not less.

"Writing code is cheap now. Many of our engineering habits are built around the constraint that it was expensive. That constraint is gone."

— Simon Willison

FURTHER READING

01. path.kilo.ai

Pragmatic guides for engineers, leads, execs

02. agents.md

Open standard for guiding coding agents

03. simonwillison.net

Deep agentic engineering patterns

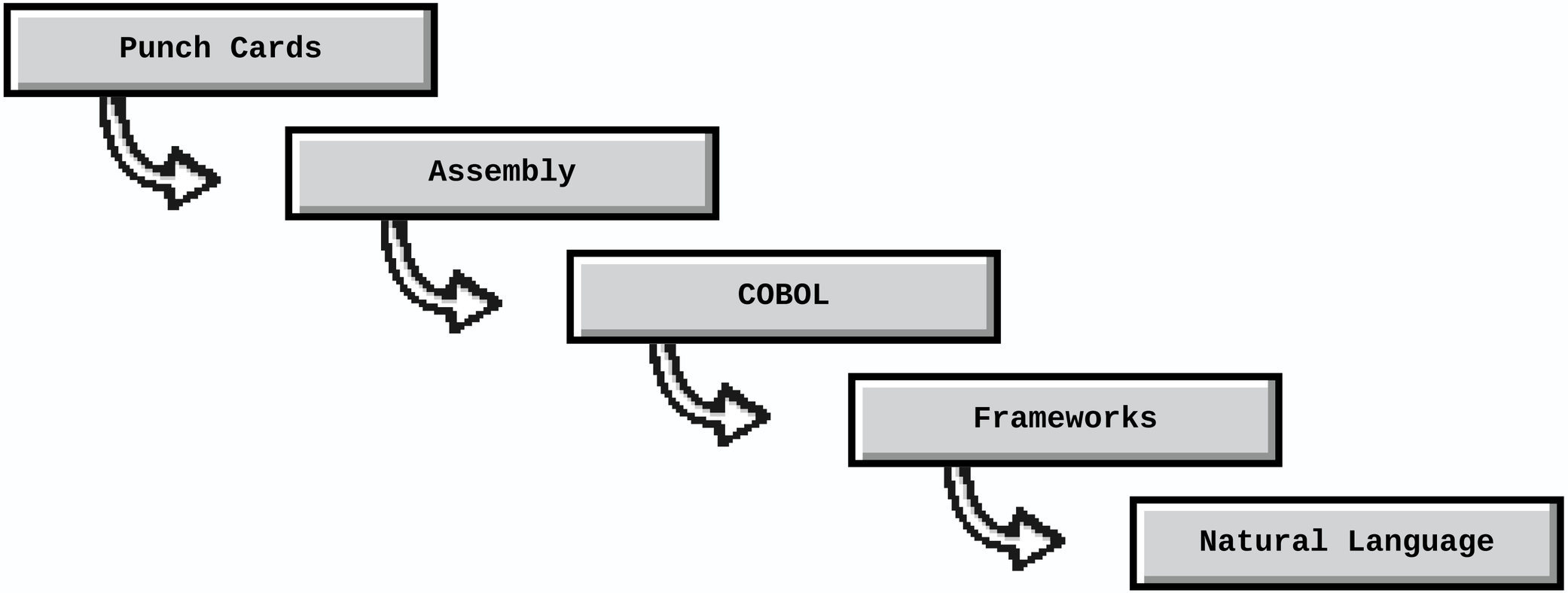
@olearycrew

AMAZING GRACE

She didn't write the first compiler so people could stop learning. She wrote it so people could learn different things — higher-level things, more valuable things.

Grace

AMAZING GRACE



Kilo Code

Home Content **Contact**

THANK YOU!

kilo.ai

@olearycrew

Resource Page

Use these design resources
in your Canva Presentation.
Happy designing!

Don't forget to delete
this page before presenting.