The Coming Commoditization of Computational Thinking & Its Impact on Computer & Data Science

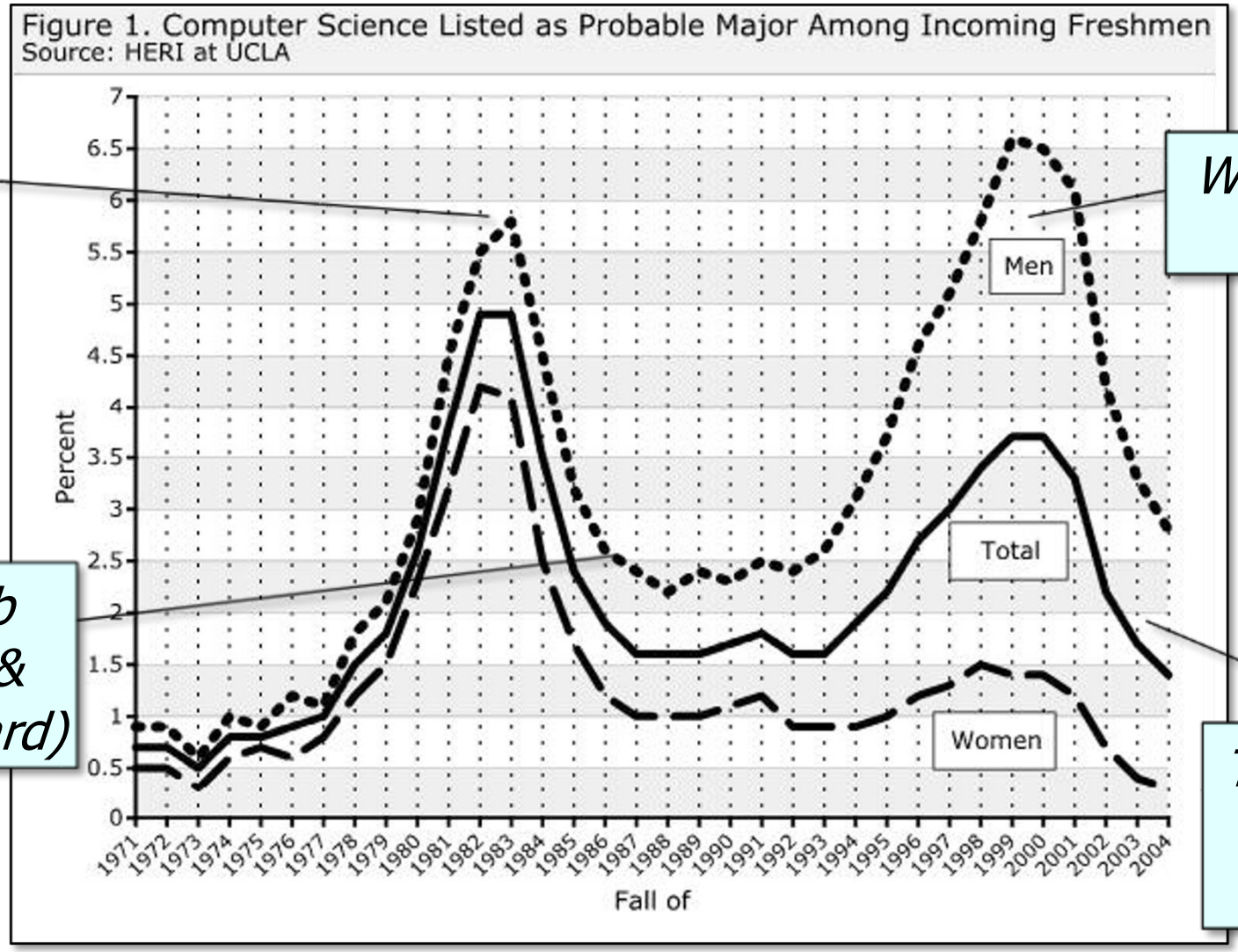
Douglas C. Schmidt
dcschmidt@wm.edu
www.cs.wm.edu/~dcschmidt

Dean of Computing,
Data Sciences & Physics

WILLIAM & MARY
CHARTERED 1693

Motivation for this Presentation

- I found a 2006 report from Microsoft Research on trends in CS enrollment



Won't have a job if not a CS major

Won't be a millionaire if not a CS major

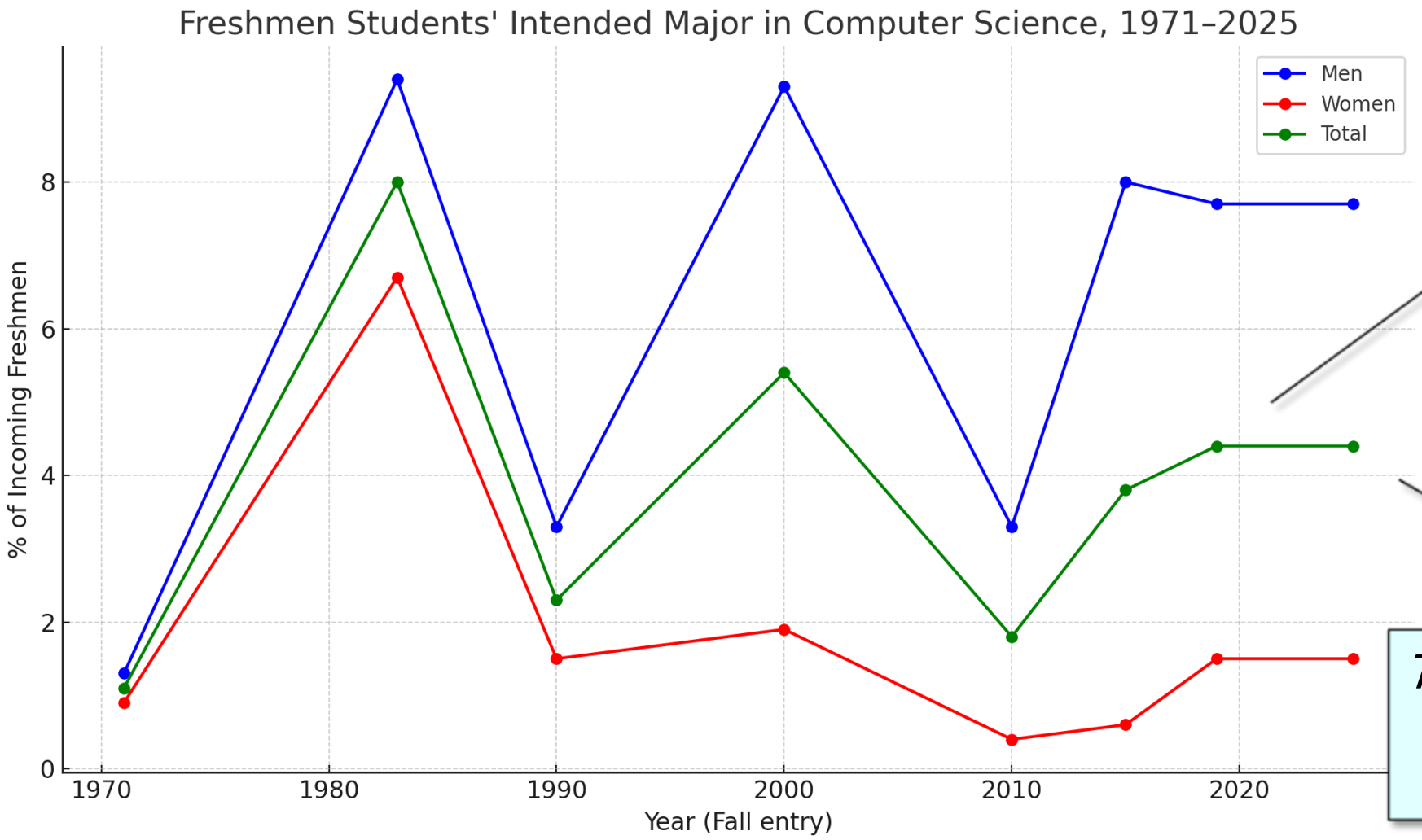
Could still get a job without CS major (& programming was hard)

The Dot Com bubble burst & outsourcing became popular

See www.cs.wm.edu/~dcschmidt/PDF/TheFutureofInformationTechnology.pdf

Motivation for this Presentation

- I was curious if I could extend this visualization over the past ~50 years



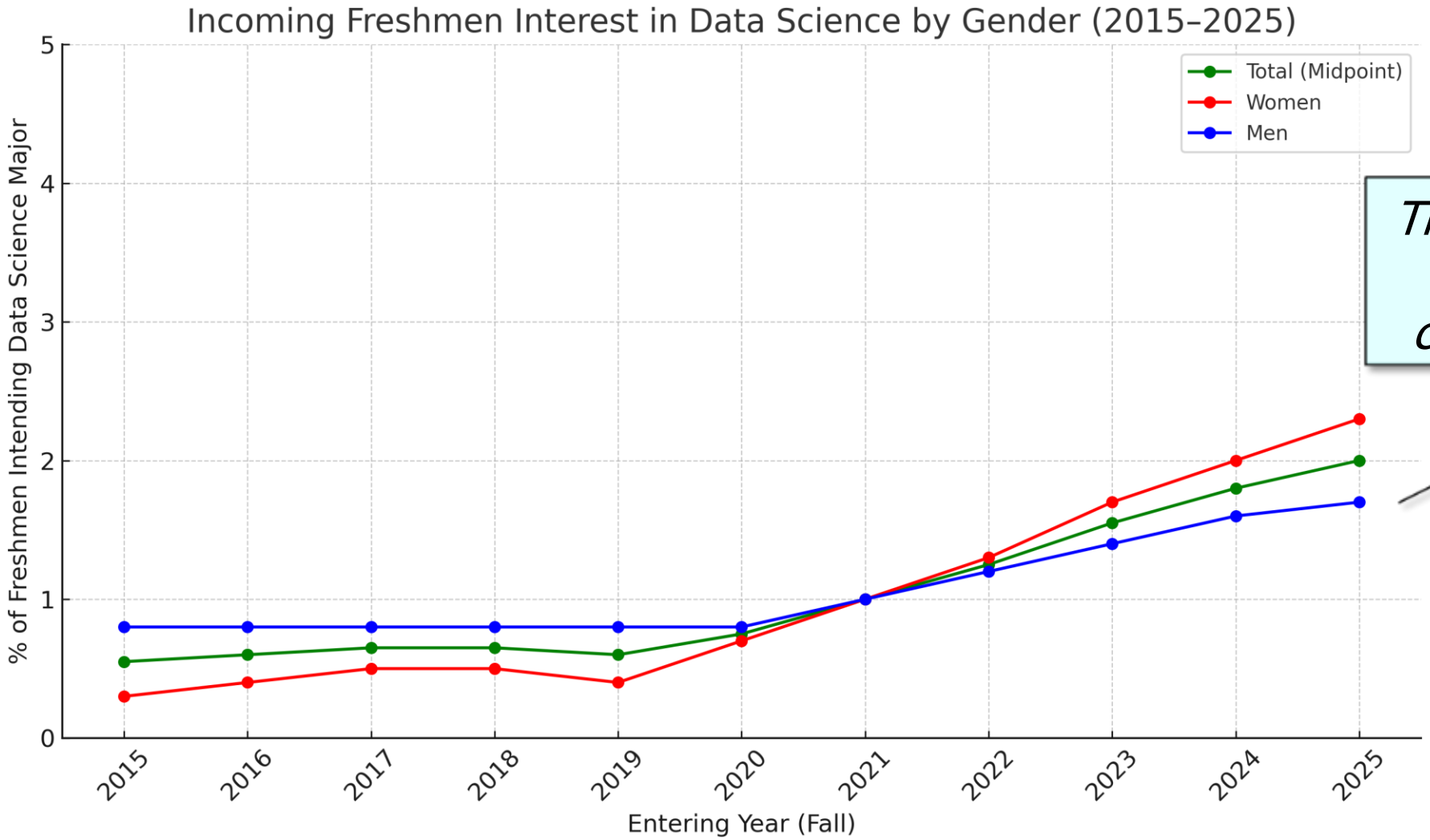
The $1.4 million dollar question is "will this trend continue"?

This graph shows a simple example of "no-code" computational thinking!

See chatgpt.com/share/6803e8fa-14e8-800d-94b5-18772a468eda

Motivation for this Presentation

- I also visualized these trends for Data Science over the past 10 years

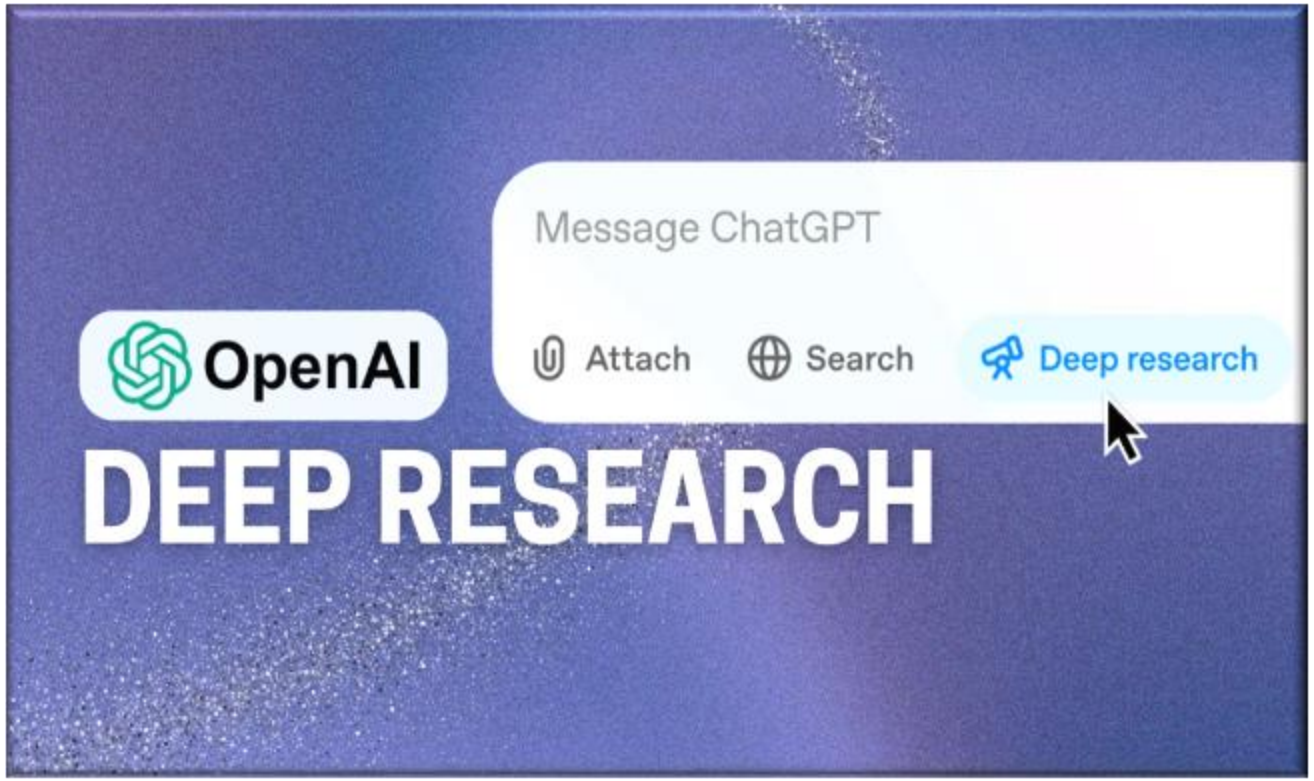


This graph also shows an example of "no-code" computational thinking!

See chatgpt.com/share/68062a5d-2d5c-800d-bef9-122ee5c1acc8

Quick Introduction to OpenAI's Deep Research

- Deep Research is an AI-powered agent that autonomously conducts multi-step research by browsing & analyzing online sources to generate detailed, citation-backed reports on complex topics

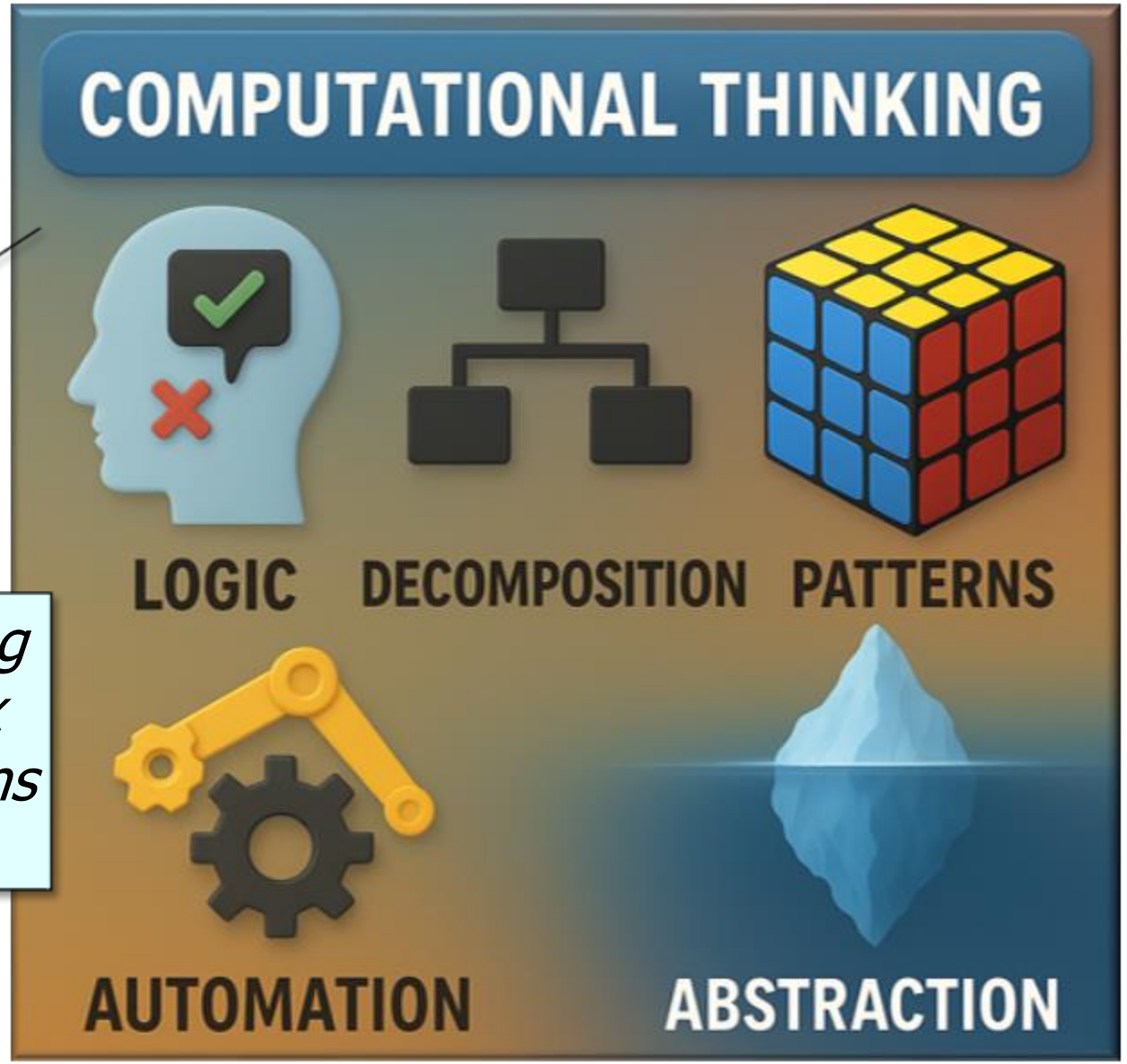


It's like having a team of talented post-docs to carry out your bidding!

See openai.com/index/introducing-deep-research

Overview of Computational Thinking

- Many disciplines can benefit from applying *computational thinking*



Computational thinking isn't about coding per se—it's a means of turning complex problems into clear, step-by-step solutions that humans or machines can execute

See en.wikipedia.org/wiki/Computational_thinking

Challenges with Computational Thinking Heretofore

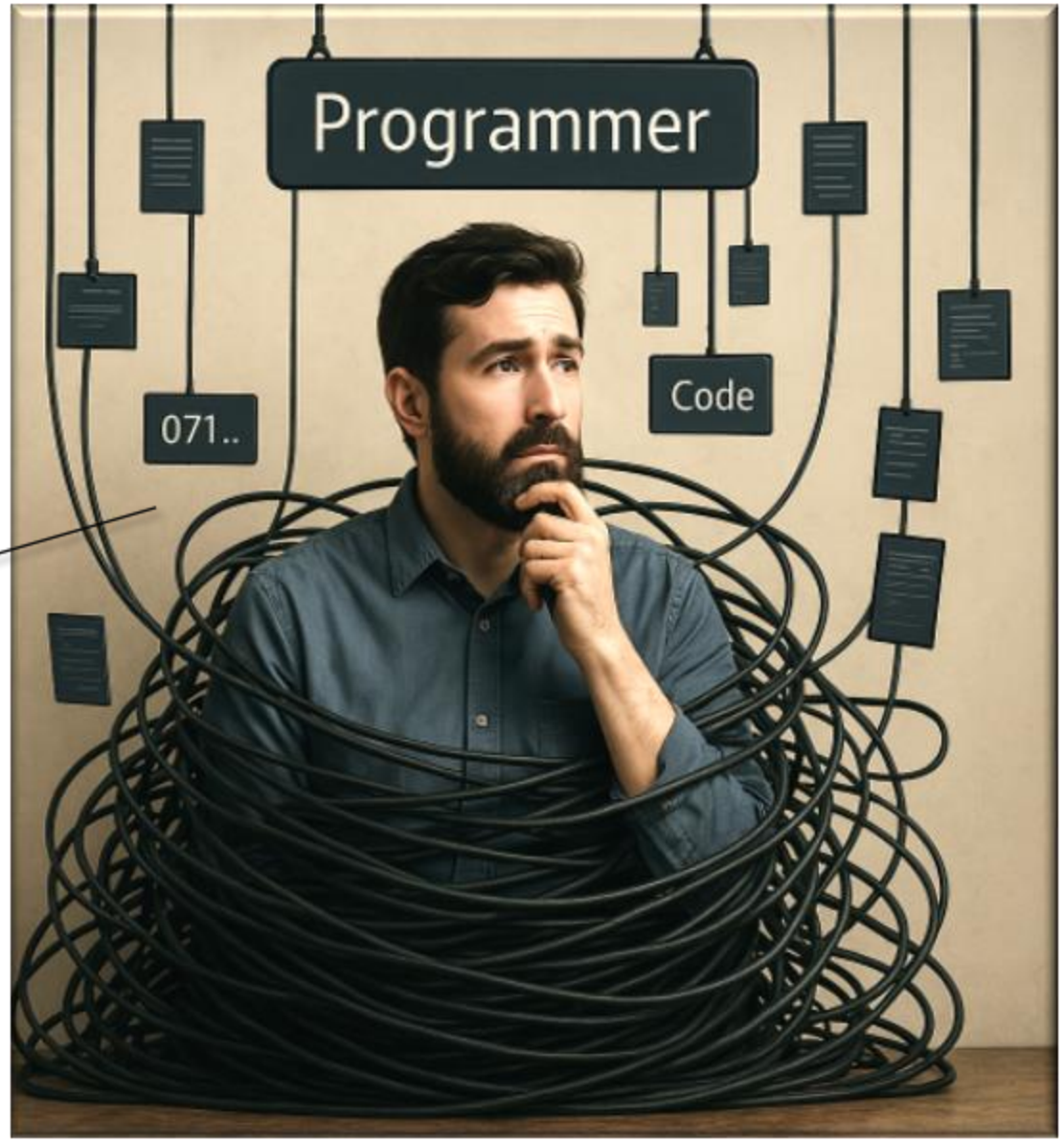
- Computational thinking historically required programming language fluency

See www.watelectronics.com/types-of-programming-languages-with-differences

Challenges with Computational Thinking Heretofore

- However, programming language fluency requires wrestling with complexities that distract from the domain to which they are applied

e.g., syntax & semantics, memory management, concurrency control, fence-post errors, buffer overflows, etc.



See dev.to/marek/why-programming-languages-are-hard-19ch

Analogy: Evolution of Automobile Technology

~120 years ago if you wanted to drive a car you had to know how to build a car

Analogy: Evolution of Automobile Technology

~70 years ago if you wanted to drive a car you had to know how to maintain a car

Analogy: Evolution of Automobile Technology

Today if you want to drive a car you only must know how to put gas in it or plug it in

Analogy: Evolution of Automobile Technology

Soon you won't even have to know how to drive a car!

Evolution of Computational Thinking

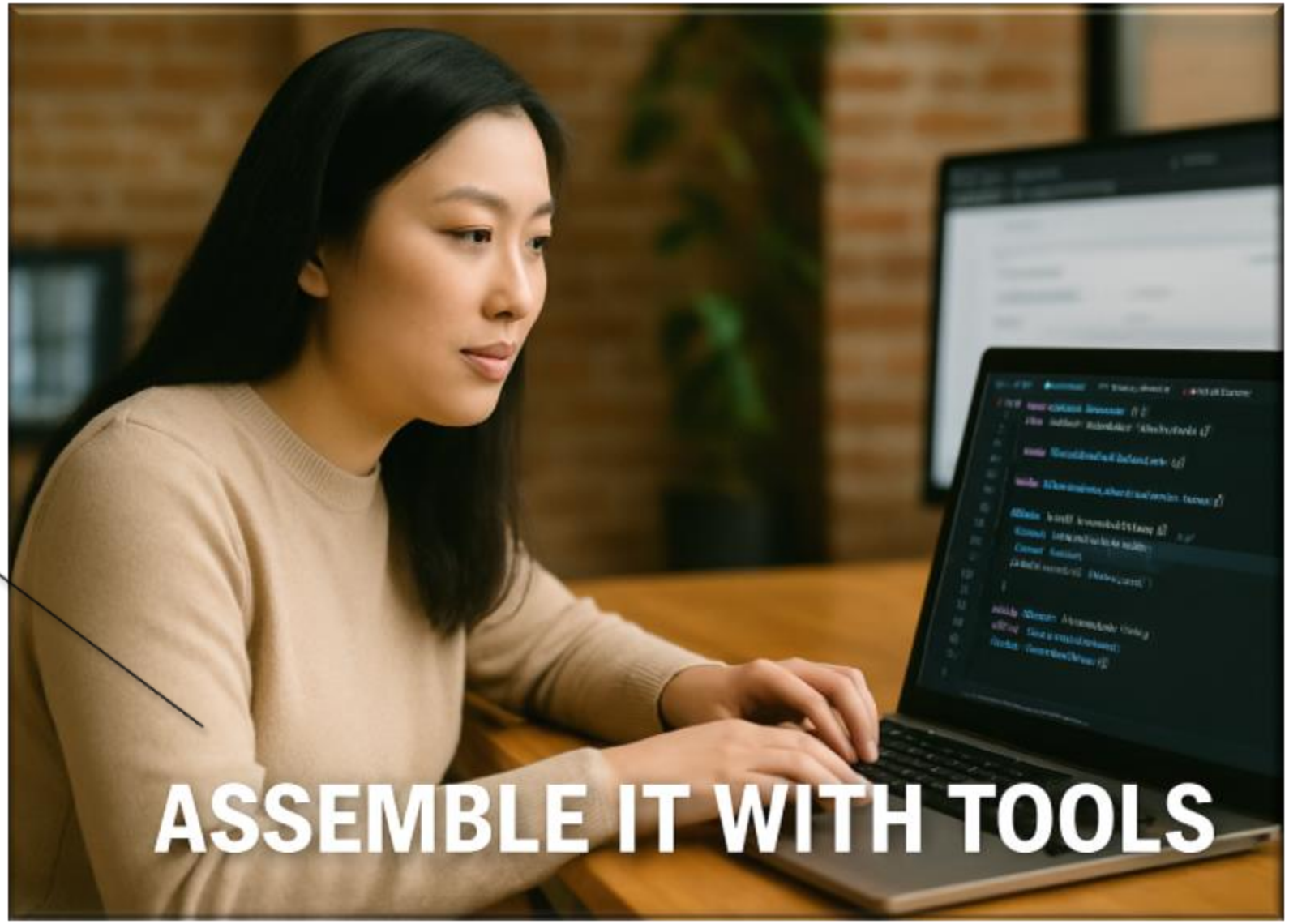
70 years ago, if you wanted to use a computer, you had to know how to build programs from scratch

Evolution of Computational Thinking

40 years ago, if you wanted to write software, you had to know how to code in a 3rd-generation language & debug low-level issues manually

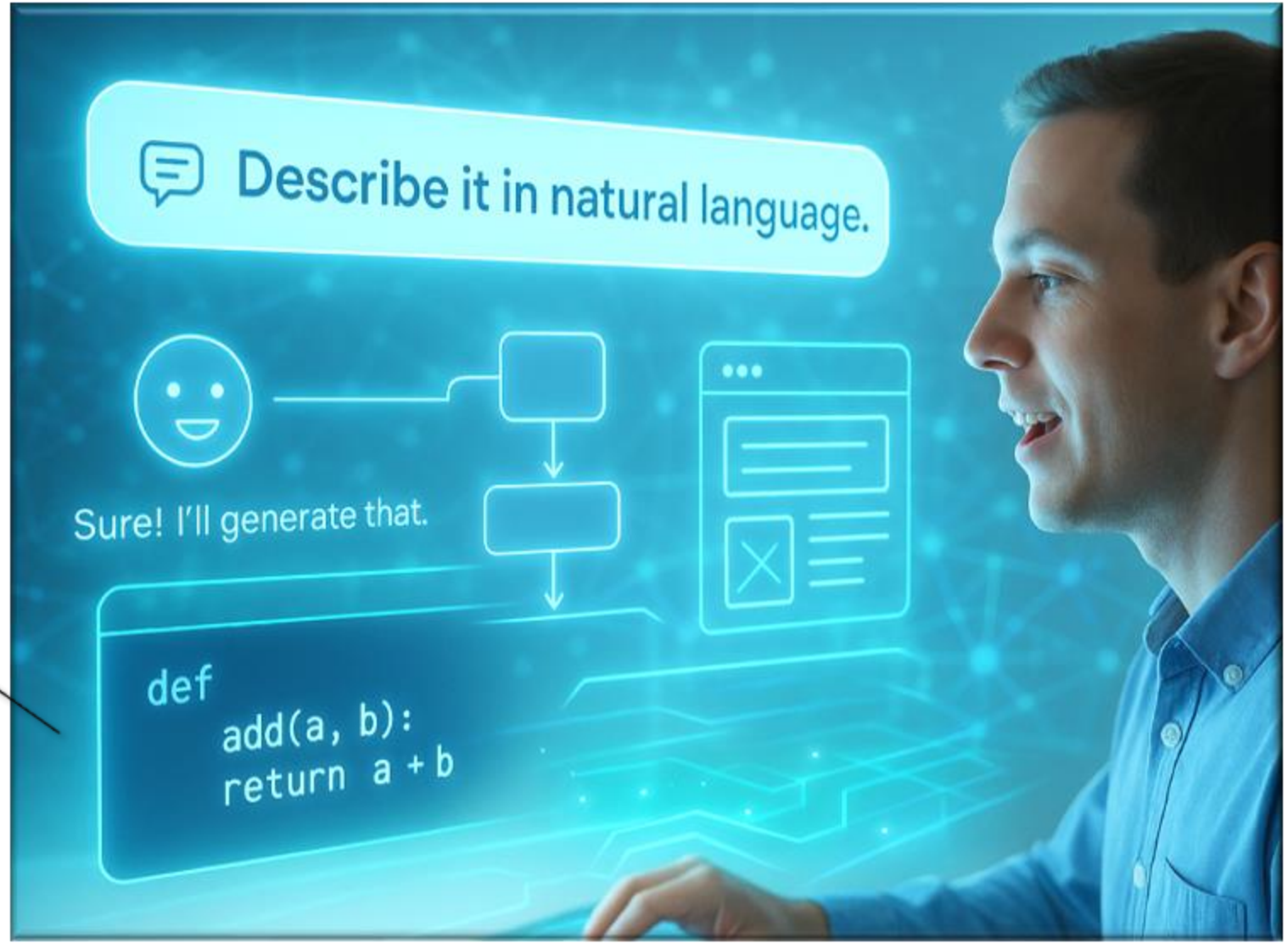
Evolution of Computational Thinking

Today, if you want to develop software, you mostly need to install the right libraries, use frameworks, & plug into APIs



Evolution of Computational Thinking

Soon, you won't even need to write much code—you'll just describe what you want in natural language, & AI will generate & adapt the solution



See windsurf.com/editor & www.cnbc.com/2025/04/16/openai-in-talks-to-pay-about-3-billion-to-acquire-startup-windsurf.html

Commoditization of Computational Thinking

- Now anyone can apply computational thinking by becoming proficient with *prompt engineering*

Involves structured interactions with LLMs to solve complex problems via natural language & prompt patterns

See en.wikipedia.org/wiki/Prompt_engineering & www.cs.wm.edu/~dcschmidt/PDF/prompt-patterns.pdf

Commoditization of Computational Thinking

- Prompt engineering lowers entry barriers to apply computational thinking in creative & technical fields

See coffeeaffection.com/most-expensive-starbucks-drink

Commoditization of Computational Thinking

- We're entering a phase where it's possible to conduct computational thinking without being proficient in traditional programming languages

Often more appropriate for non-computing professionals that traditional programming

See youtu.be/OMvuRtZNm08 for a demo of computational thinking with early ChatGPT

Commoditization of Computational Thinking

- We're entering a phase where it's possible to conduct computational thinking without being proficient in traditional programming languages

See youtu.be/OMvuRtZNm08 for a demo of computational thinking with early ChatGPT

Advanced Data Analysis

D

Here's input for 35 faculty members and their top 3 topic preferences ordered from highest to lowest out of a total of 5 topics (numbered 1, 2, 3, 4, 5). Please generate output that allocates all 35 faculty members to all 5 topics, balancing them evenly (7 faculty per topic) taking their ordered preferences into account. Also, please give preference to faculty in the order they appear in the list below, which has the following example format:

Doug Schmidt [3,5,1]

which means Doug Schmidt's top preference is 3, followed by 5, followed by 1.

Here is the list of faculty and their preferences in the format described above:

Gautam Biswas [3,5,1]
Aniruddha Gokhale [3,2,1]

Regenerate

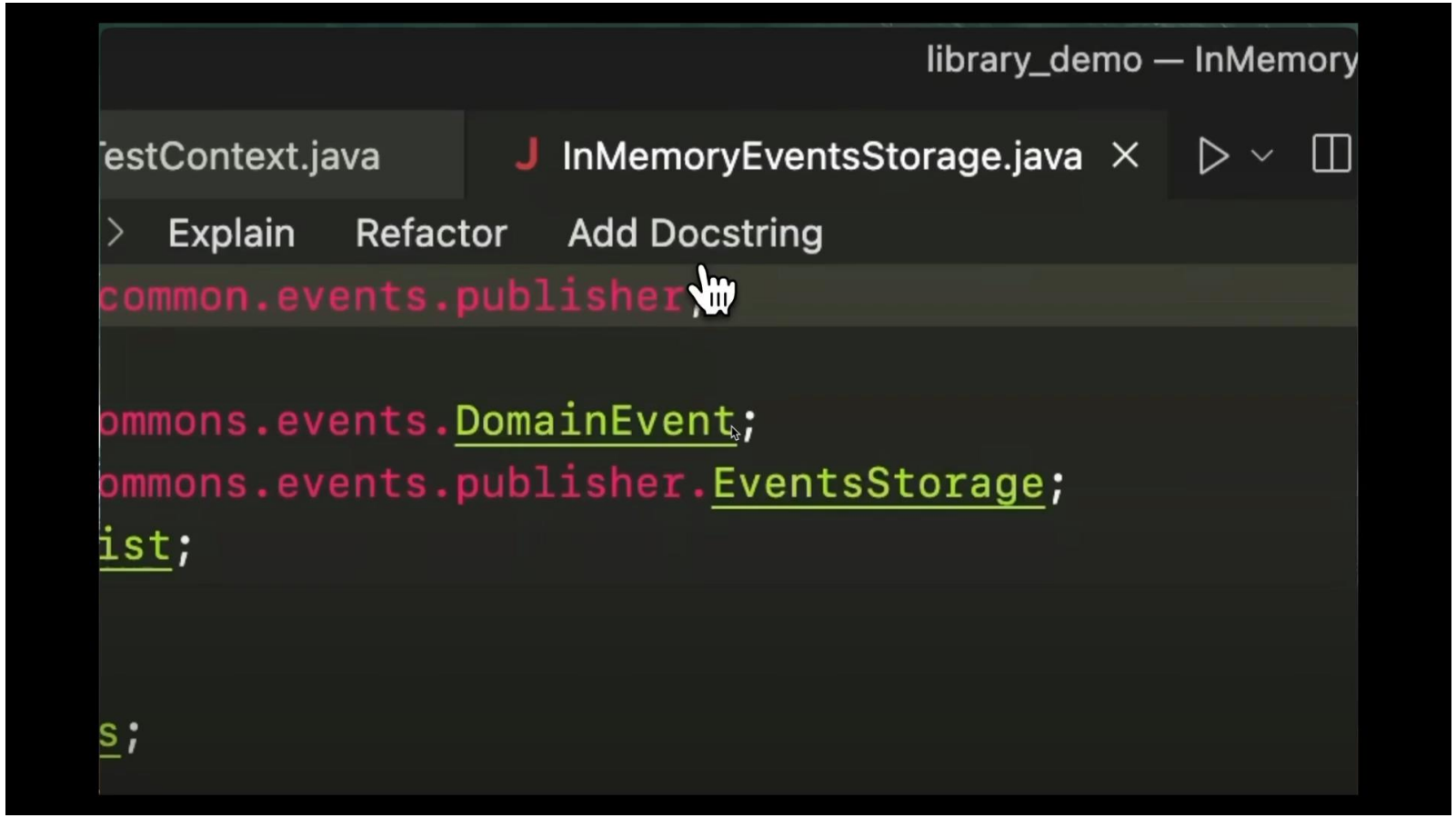
Send a message

ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT August 3 Version
To generate more accurate information based on recent facts, use LINER. LINER Search

See chatgpt.com/c/e8618440-15db-43d7-ab61-4916725f2494 for the chat session

Natural Language as the New "Programming" Model

- LLMs embedded in IDEs can now translate natural language into code
  - e.g., Python, Java SQL, HTML, even LaTeX, etc.



windsurf

See analyticsindiamag.com/ai-news-updates/cursor-github-copilot-rival-codeium-launches-windsurf-first-agentic-ide-for-coding

Natural Language as the New "Programming" Model

- Tools like ChatGPT, Claude, Copilot, & Code Interpreter allow computational thinkers to:
  - Analyze data
  - Generate simulations
  - Write scripts
  - Build interactive tools

AI bridges the gap between idea & implementation

Computational Thinking Use Cases Across Disciplines

- **CS Faculty**: Mapping professors to discussion topic groups
- **Humanities**: Generate stylometric analysis of Shakespeare's plays
- **Biology**: Perform k-means clustering on protein data
- **Economics**: Simulate supply chain disruptions in Python
- **Sociology**: Summarize massive datasets into trends & narratives

All these use cases can be realized robustly without writing a single line of traditional code

Computational Thinking Use Cases Across Disciplines

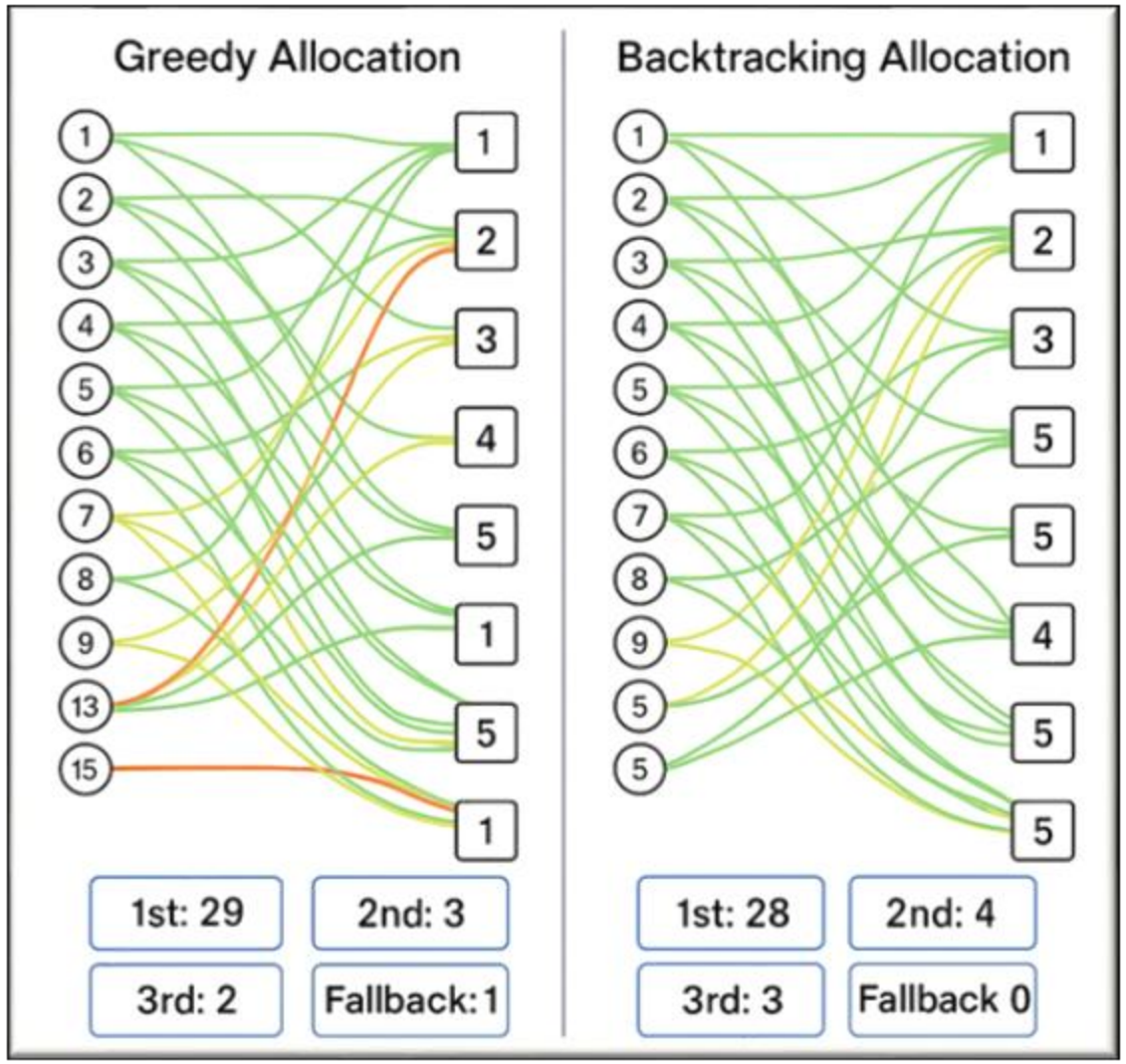
- **CS Faculty**: Mapping professors to discussion topic groups

Here's input for 35 faculty members and their top 3 topic preferences ordered from highest to lowest out of a total of 5 topics (numbered 1, 2, 3, 4, 5). Please generate output that allocates all 35 faculty members to all 5 topics, balancing them evenly (7 faculty per topic) taking their ordered preferences into account. Also, please give preference to faculty in the order they appear in the list below, which has the following example format:

Doug Schmidt [3,5,1]

which means Doug Schmidt's top preference is 3, followed by 5, followed by 1.

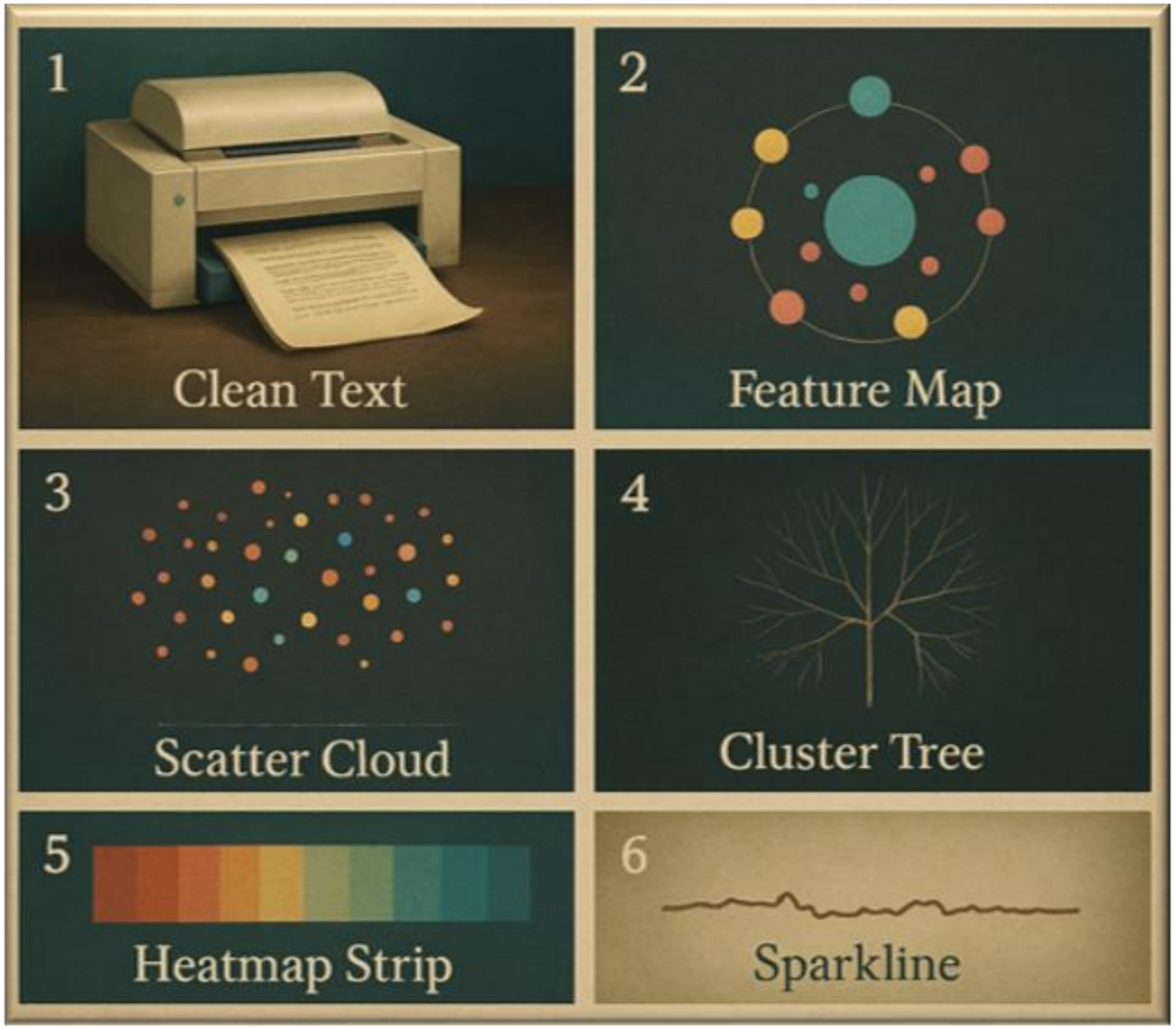
Here is the list of faculty and their preferences in the format described above:



See chatgpt.com/share/68044851-1734-800d-a03b-3d54e529f216 to see computational thinking with recent ChatGPT

Computational Thinking Use Cases Across Disciplines

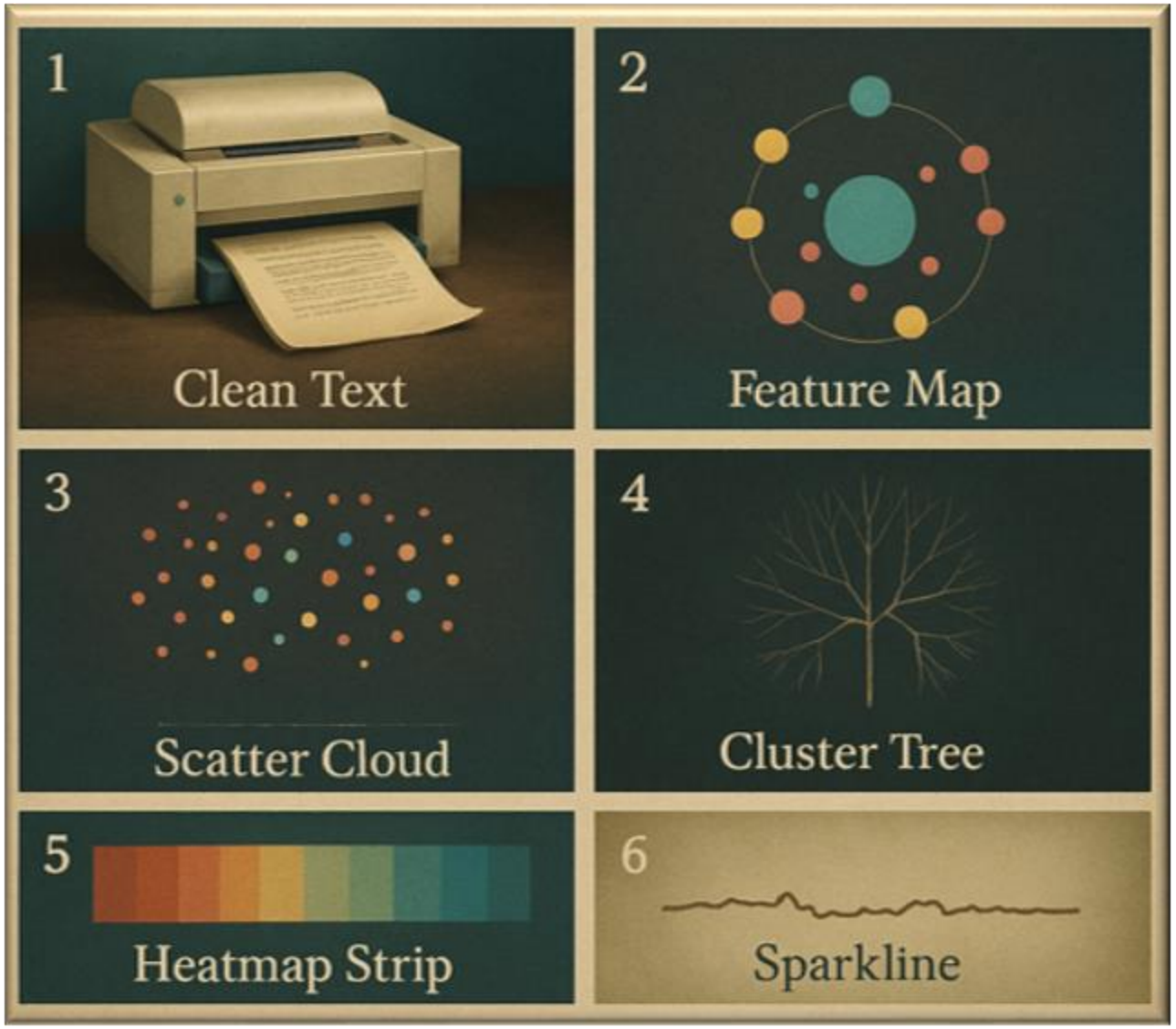
- **Humanities**: Generate stylometric analysis of "Shakespeare's" plays



See en.wikipedia.org/wiki/Stylometry

Computational Thinking Use Cases Across Disciplines

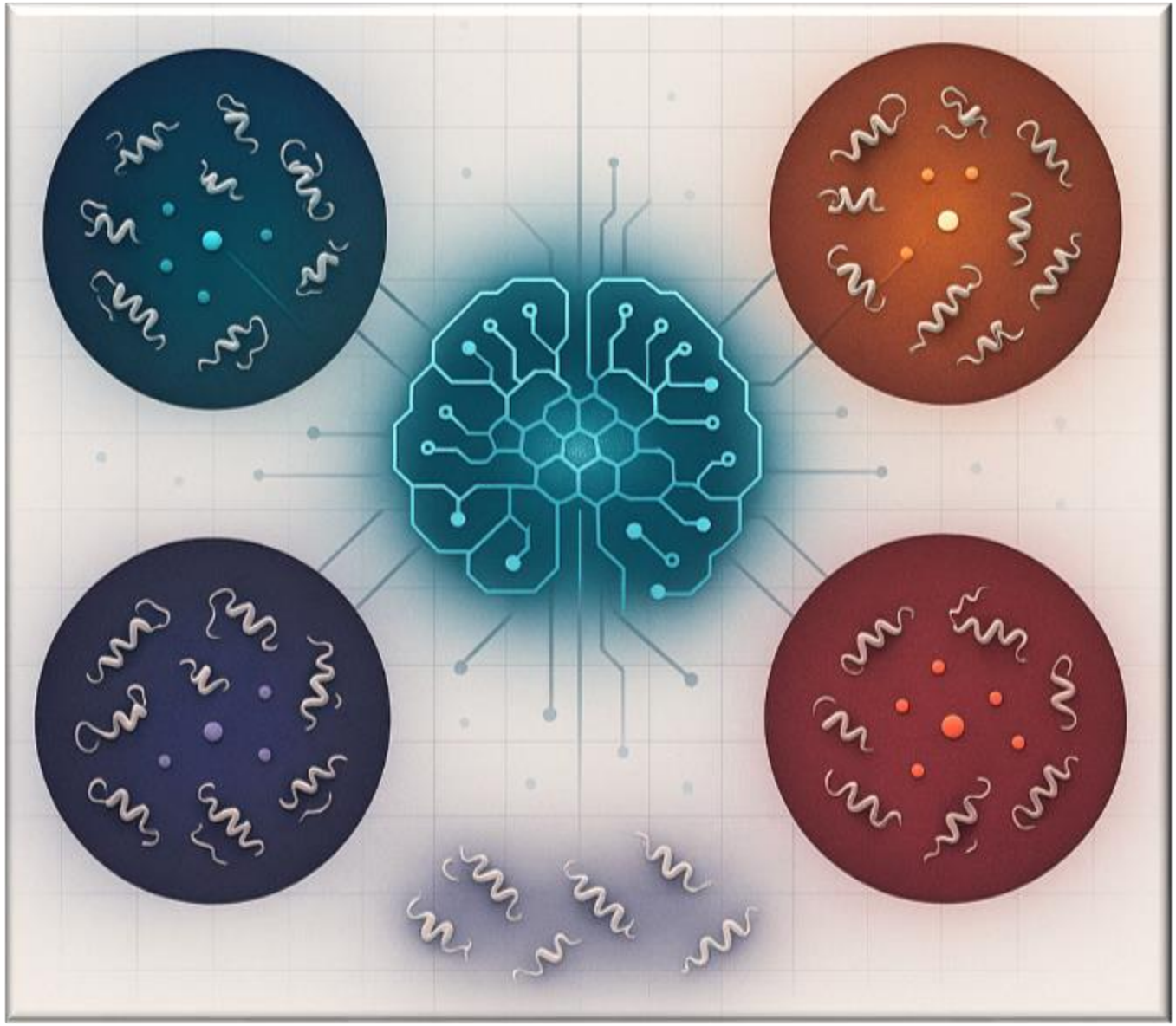
- **Humanities**: Generate stylometric analysis of "Shakespeare's" plays
  - Useful to determine authorship..



See en.wikipedia.org/wiki/Shakespeare_authorship_question

Computational Thinking Use Cases Across Disciplines

- **Biology**: Perform k-means clustering on protein data
  - Identify functional groupings, anomalies, or targets for drug discovery



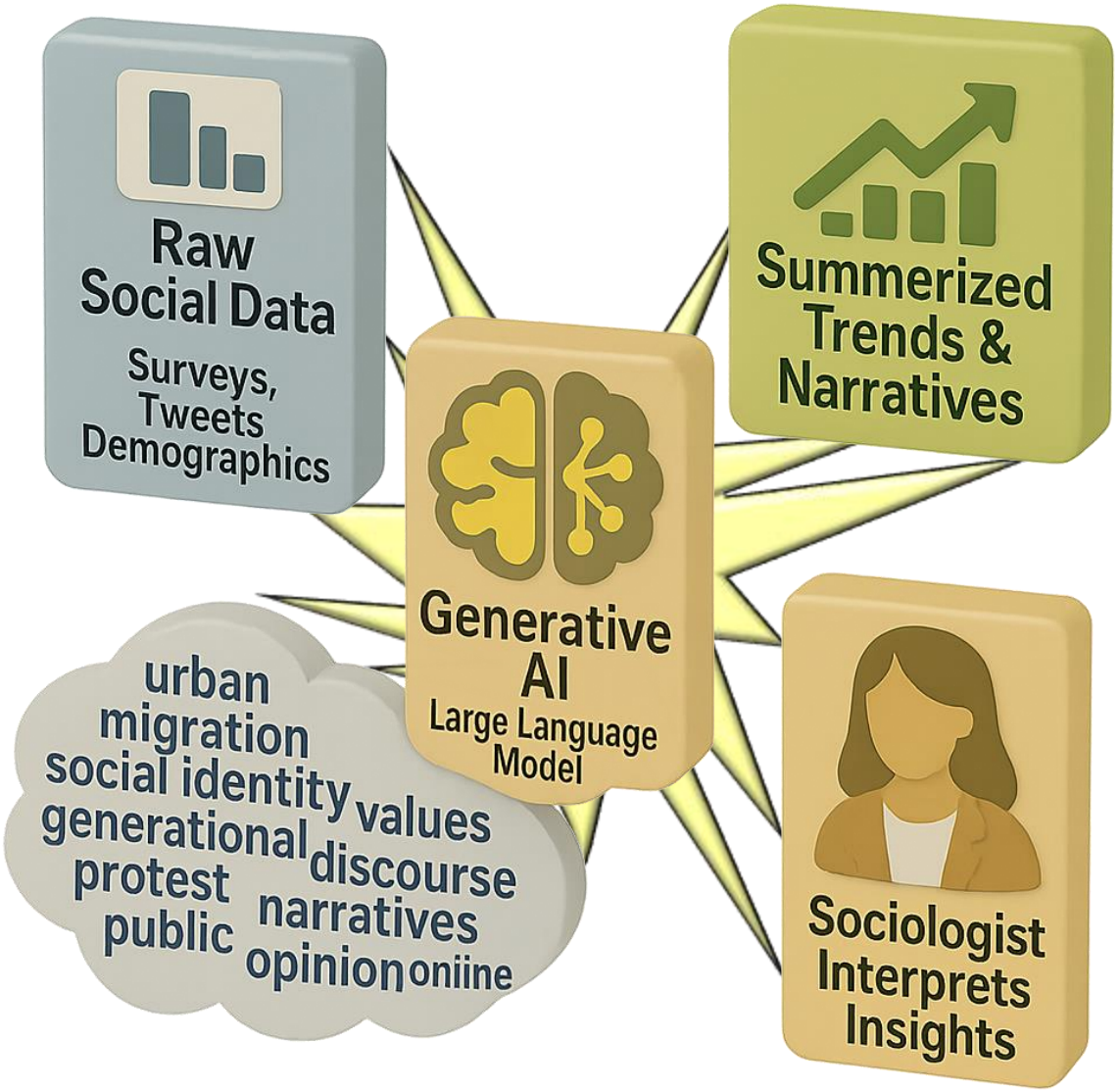
See www.biorxiv.org/content/10.1101/2023.06.14.544984v2.full.pdf

Computational Thinking Use Cases Across Disciplines

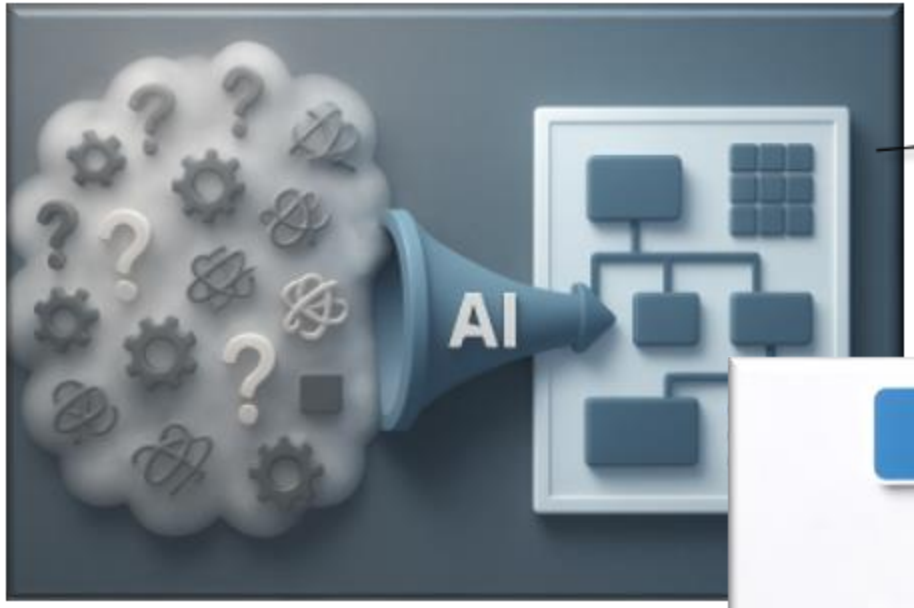
- **Economics**: Simulate supply chain disruptions
  - Generate Python simulation from prompts

Computational Thinking Use Cases Across Disciplines

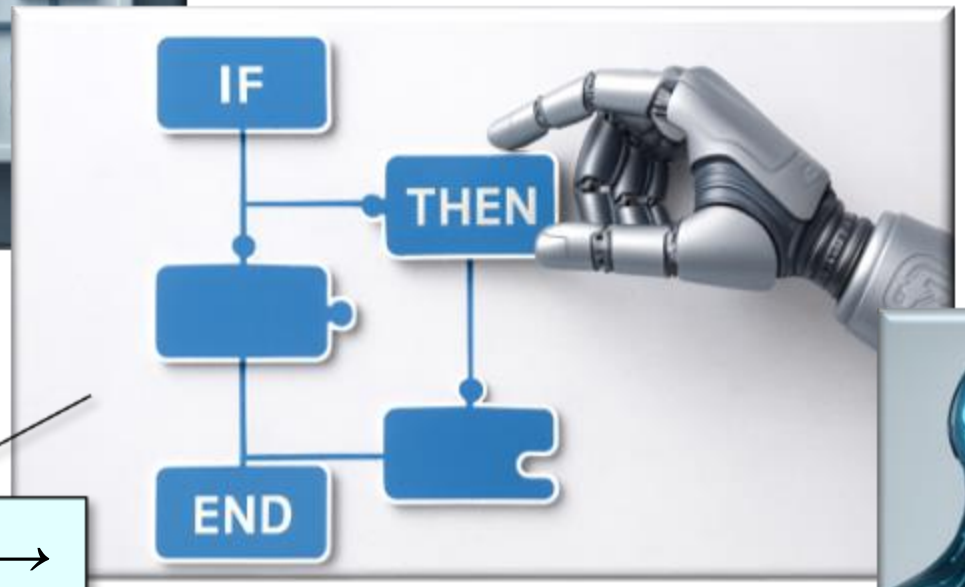
- **Sociology**: Summarize massive (& messy) datasets
  - Raw data is processed by GenAI to identify patterns, extract themes & construct narratives
  - A sociologist analyzes AI-generated insights, adding domain knowledge & critical interpretation



Three Layers of Computational Thinking Being Commoditized



1. Problem abstraction → AI helps translate vague problems into structured prompts



3. Tool implementation → AI generates actual code or analysis

2. Automation design → AI scaffolds algorithms in natural language

Knowledge work is becoming computational at scale

Rethinking the Computer Science Curricula

- Should everyone still learn to code?
  - Probably—but not for the same reasons or in the same way

Rethinking the Computer Science (or Data Science) Curricula

- Focus should shift to teaching students to collaborate with AI

Designing good prompts

Interpreting AI-generated solutions

Verifying outputs & building on them

Teaching computational thinking as a literacy, not just a technical skill

Rethinking the Computer Science (or Data Science) Curricula

- Focus should shift to teaching students to collaborate with AI, e.g.
  - Teach them how to craft thoughtful prompts

Designing good prompts

Interpreting AI-generated solutions

Verifying outputs & building on them

Teaching computational thinking as a literacy, not just a technical skill

Rethinking the Computer Science (or Data Science) Curricula

- Focus should shift to teaching students to collaborate with AI, e.g.
  - Teach them how to craft thoughtful prompts
  - Critically interpret & verify AI-generated outputs, &

Designing good prompts

Interpreting AI-generated solutions

Verifying outputs & building on them

Teaching computational thinking as a literacy, not just a technical skill

Rethinking the Computer Science (or Data Science) Curricula

- Focus should shift to teaching students to collaborate with AI, e.g.
  - Teach them how to craft thoughtful prompts
  - Critically interpret & verify AI-generated outputs, &
  - Apply computational thinking as a foundational literacy

Designing good prompts

Interpreting AI-generated solutions

Verifying outputs & building on them

Teaching computational thinking as a literacy, not just a technical skill

Commoditized ≠ Cheapened

- Oversimplification risks
  - e.g., AI may hallucinate logic if not used properly

See en.wikipedia.org/wiki/Hallucination_(artificial_intelligence)

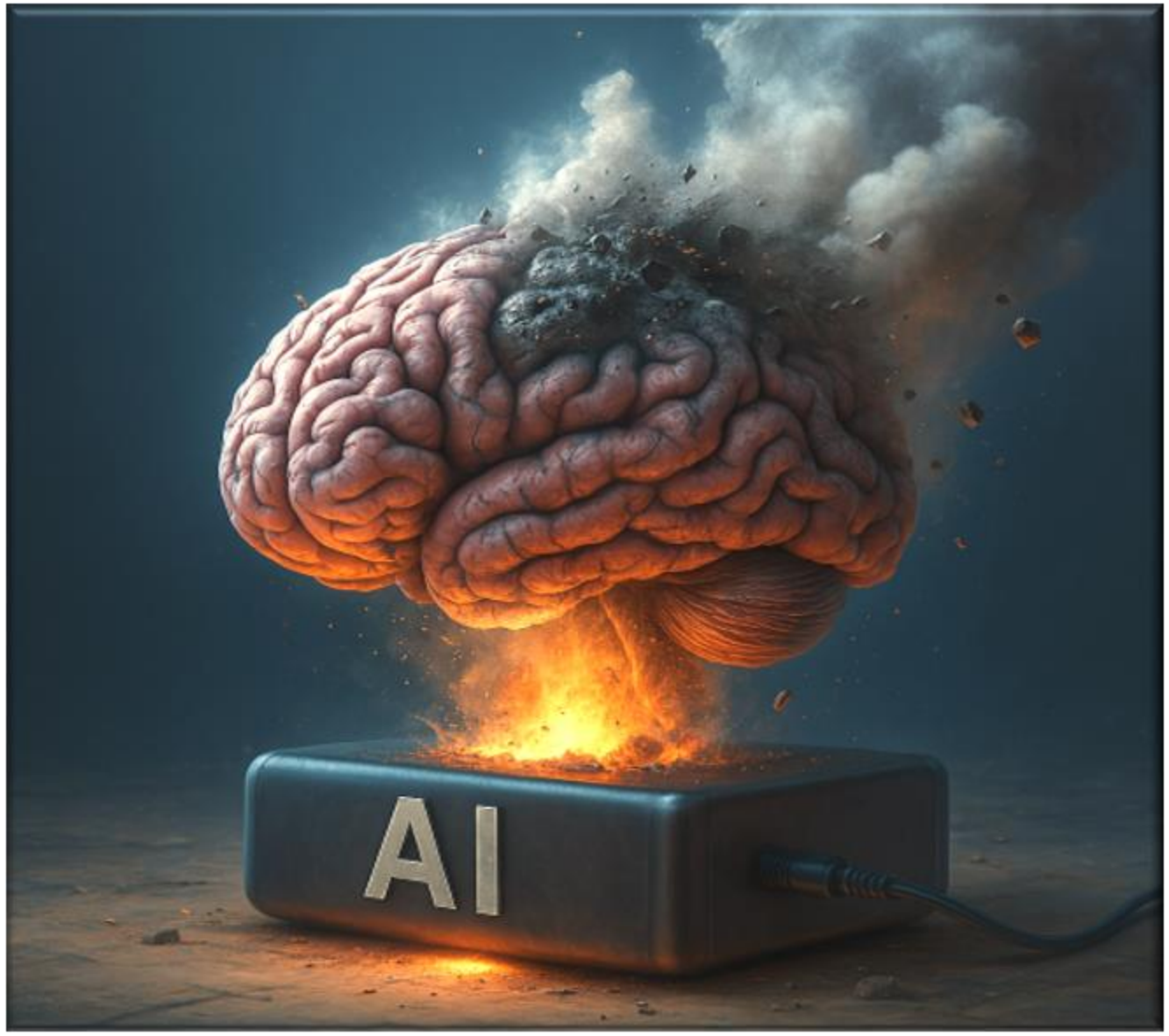
Commoditized ≠ Cheapened

- Need for meta-cognitive skills
  - Know when to trust & when to intervene so that AI doesn't outweigh human oversight

Computational thinking is easier to access, but still requires critical thinking & rigor

Commoditized ≠ Cheapened

- Need for meta-cognitive skills
  - Know when to trust & when to intervene so that AI doesn't outweigh human oversight
  - Computational thinking is easier to access, but critical thinking & rigor are still required to avoid "brain-rot"



See www.digitalinformationworld.com/2024/12/oxford-names-brain-rot-as-2024s-word-of.html

Commoditized ≠ Cheapened

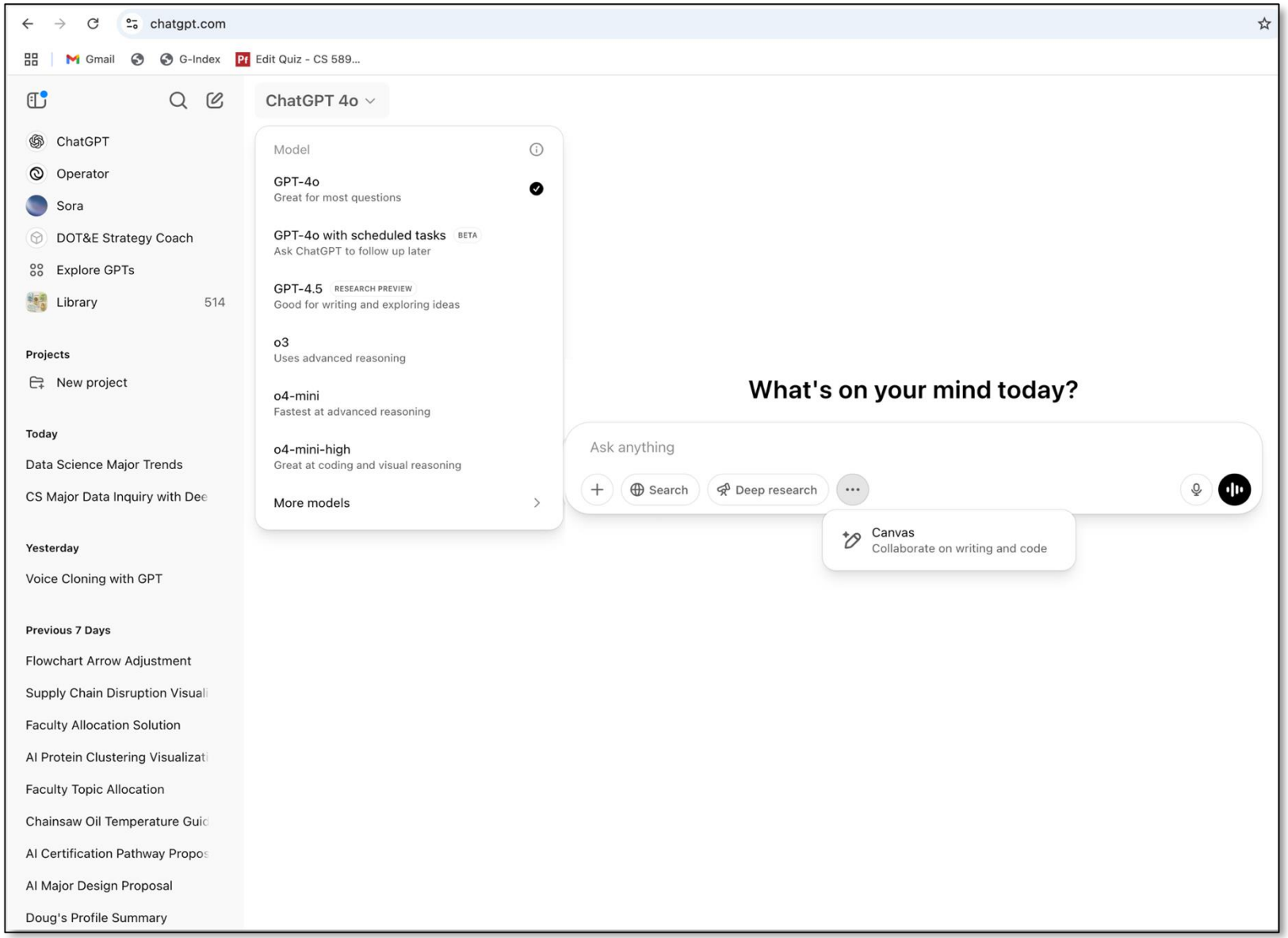
- Ethical issues
  - e.g., authorship, bias, misuse, etc.

AI copyright lawsuit: What does it mean for the future of generative AI?

See crsreports.congress.gov/product/pdf/LSB/LSB10922

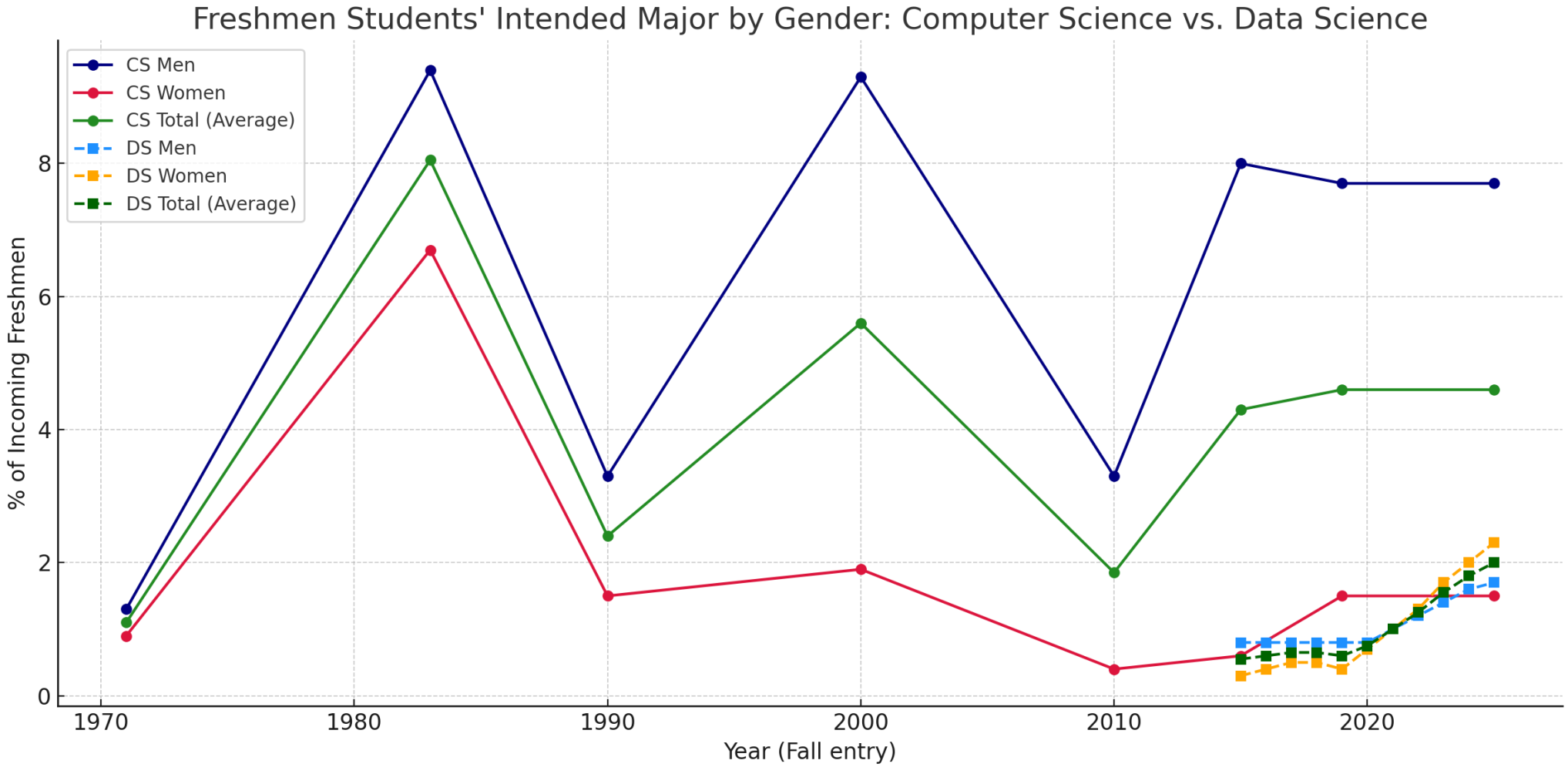
Computational Thinking is Becoming a Platform

- Tools are evolving rapidly to support computational thinking usable by many
  - Just like spreadsheets, search, & databases



Computational Thinking is Becoming a Platform

- The new "computing literacy" is not coding per se



It’s the ability to translate human questions into computable form that supports computational thinking

Computational Thinking is Becoming a Platform

- The future belongs to those who can ask the right questions & use AI to model them computationally

1. **Problem abstraction**

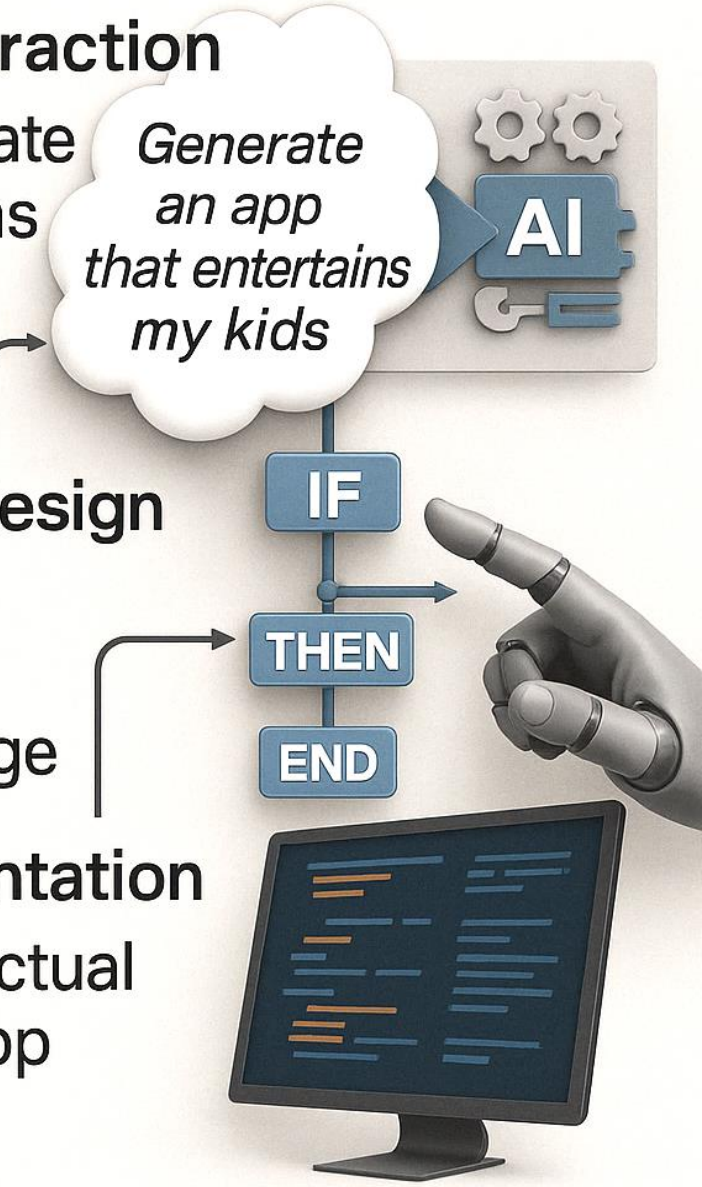
   AI helps translate vague problems into structured prompts

2. **Automation design**

   AI scaffolds algorithms in natural language

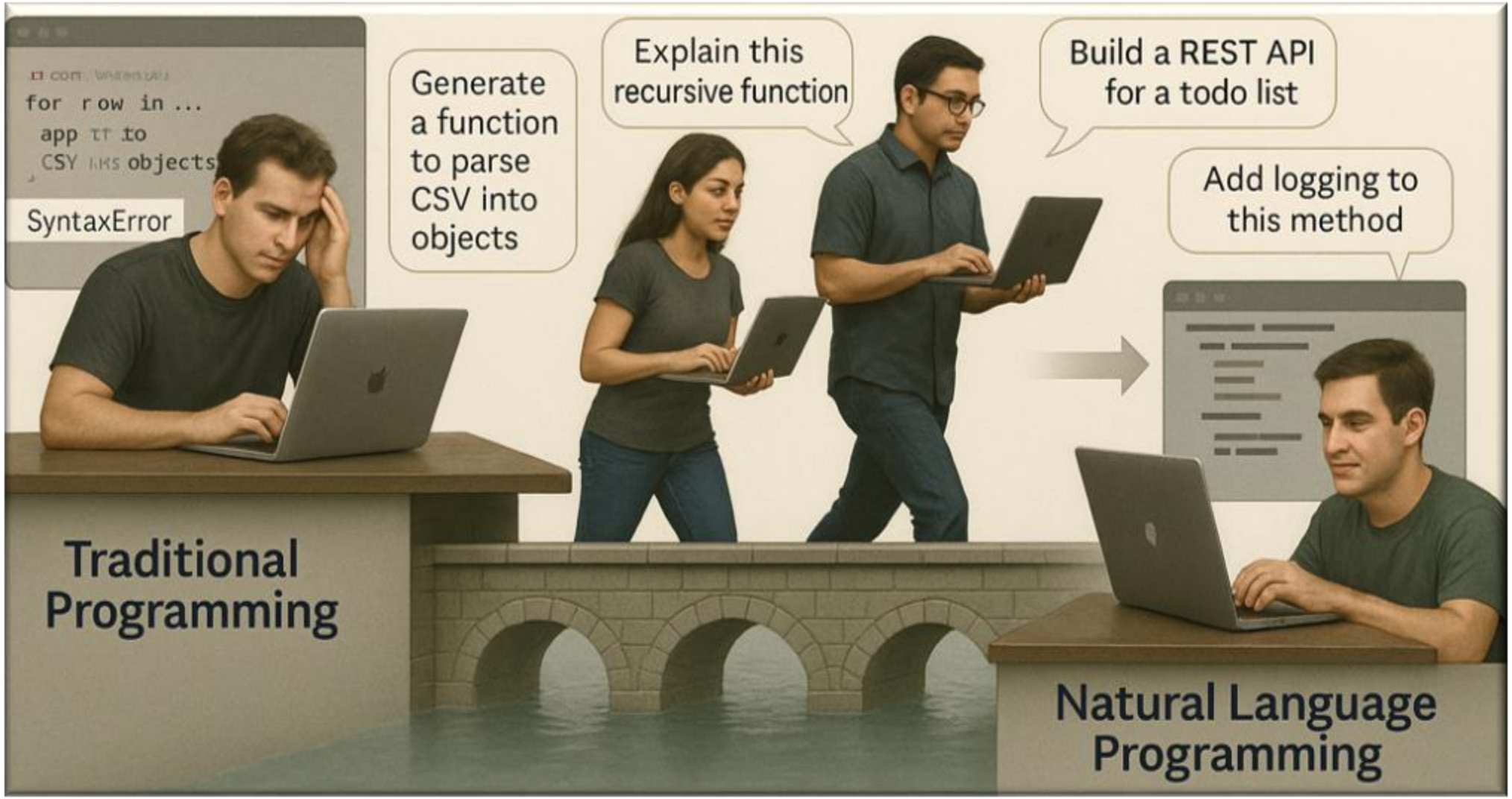
3. **Tool implementation**

   AI generates actual code for the app



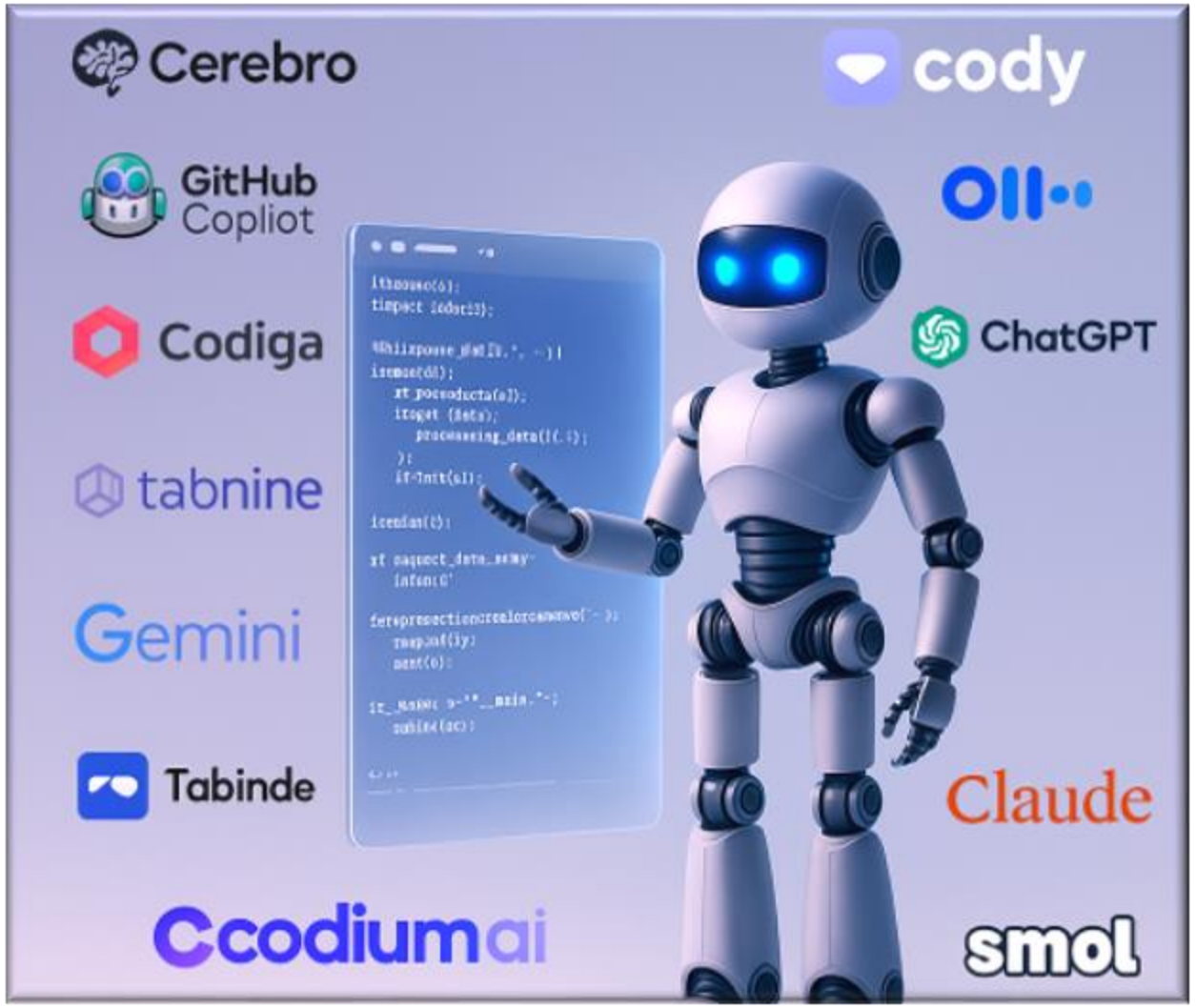
What We Can Do to Prepare Our Students (& Ourselves) for AI

- Embrace natural language programming in certain assignments



What We Can Do to Prepare Our Students (& Ourselves) for AI

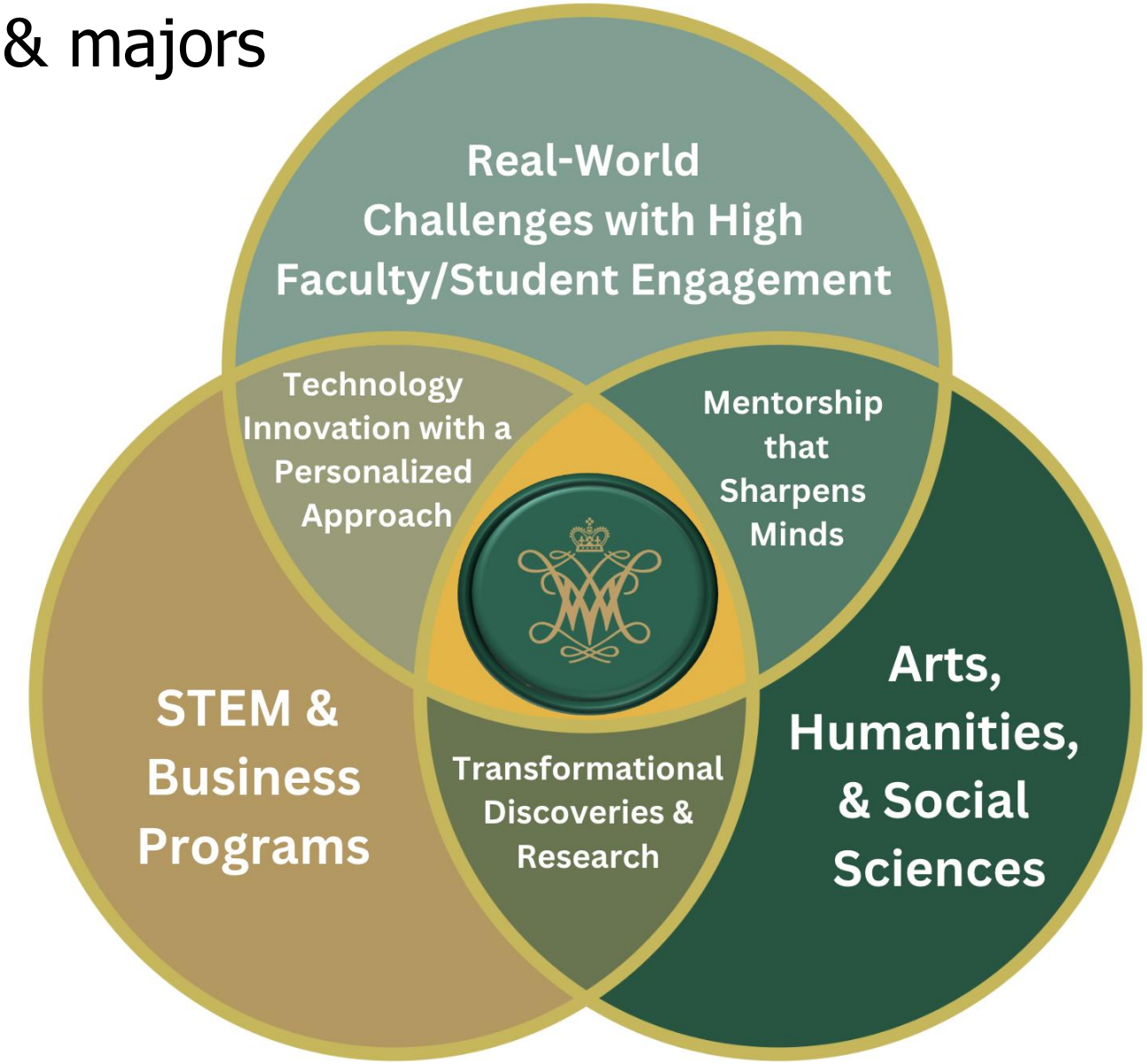
- Embrace natural language programming in certain assignments
  - e.g., train students to use the latest/greatest AI-enabled IDEs



See www.aifalabs.com/blog/best-ai-coding-tools

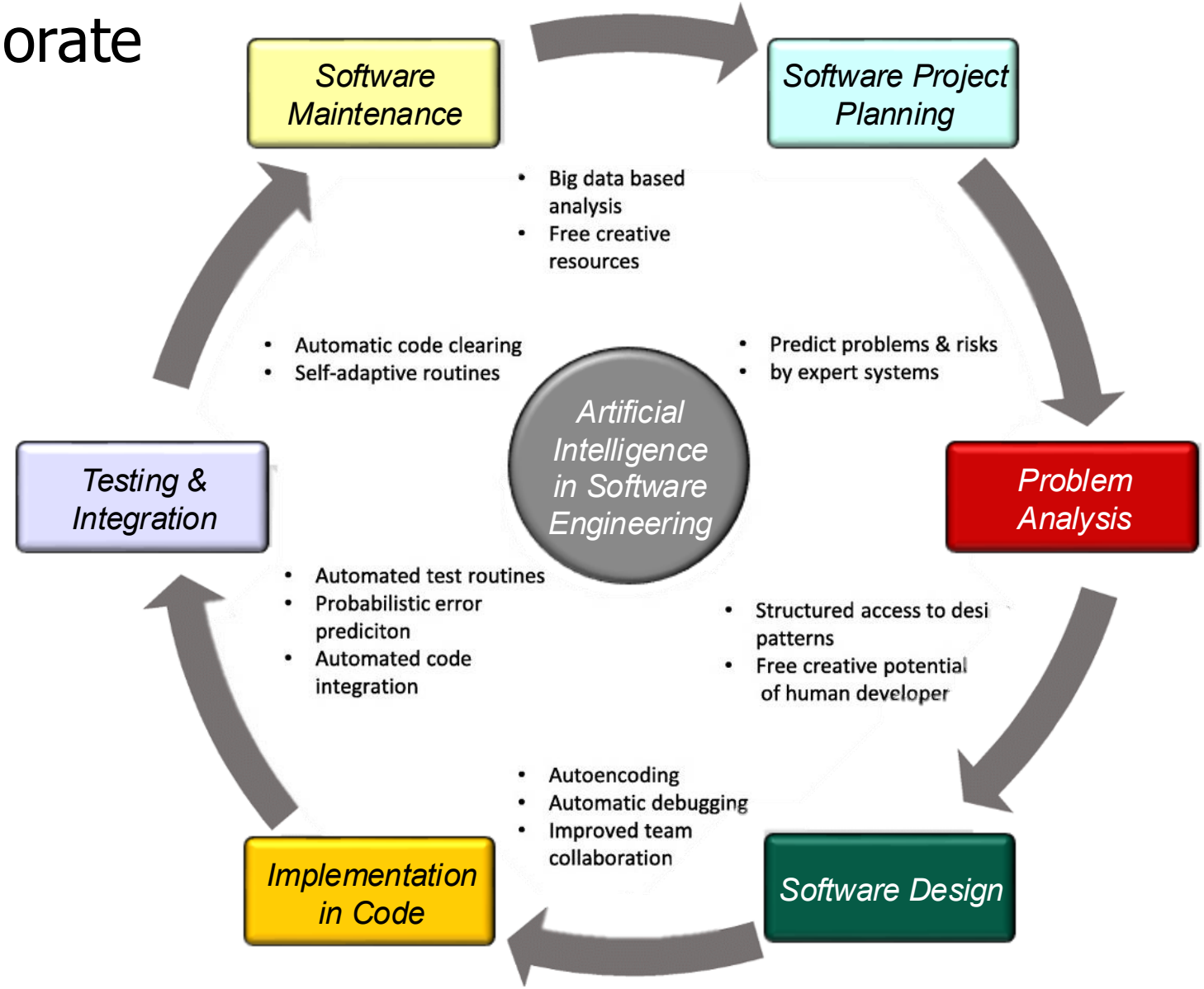
What We Can Do to Prepare Our Students (& Ourselves) for AI

- Encourage students across schools & majors to think algorithmically
  - e.g., make it easier to major or minor in CS



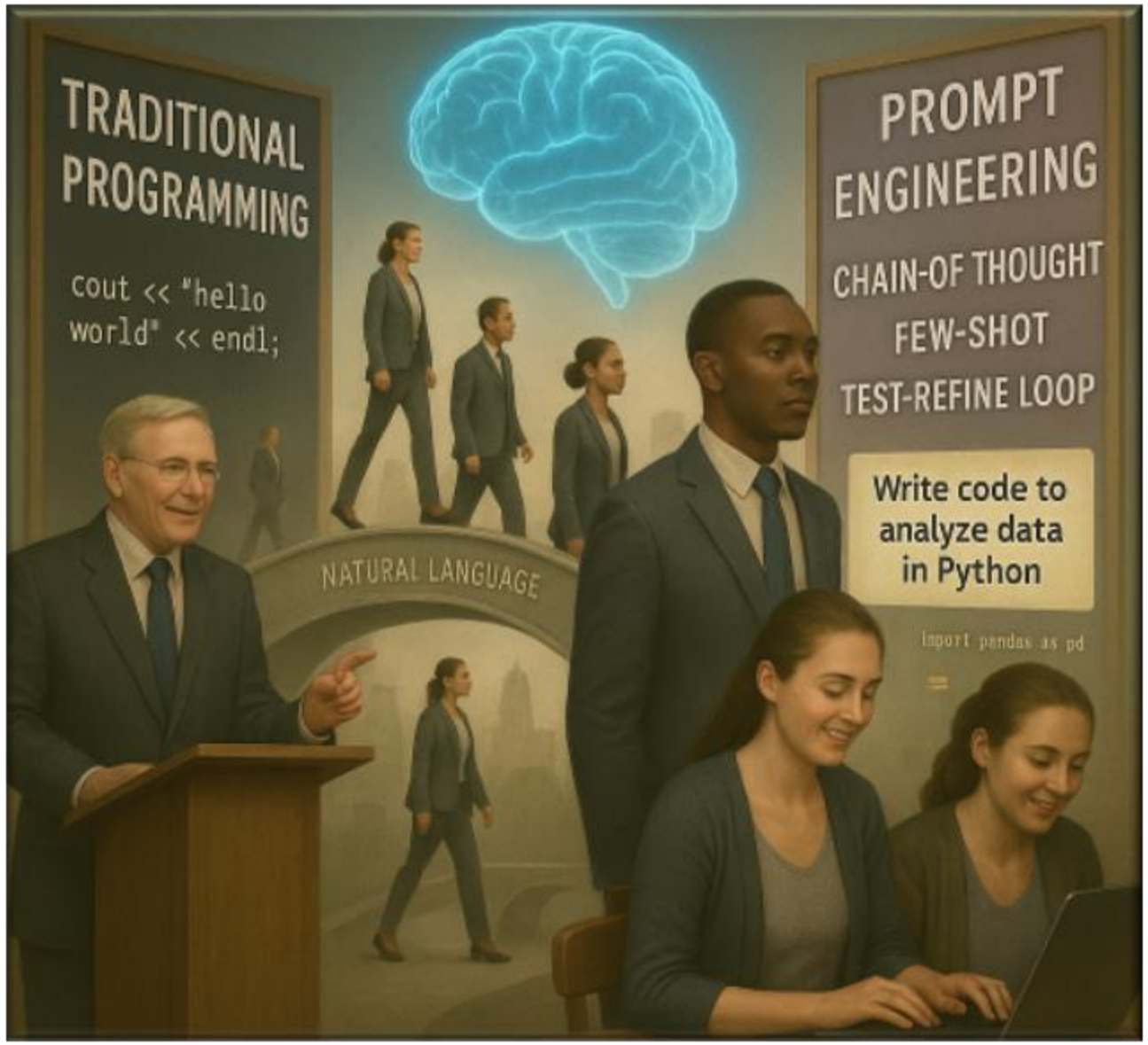
What We Can Do to Prepare Our Students (& Ourselves) for AI

- Teach students how to collaborate with AI, not compete with it
  - This can be explored both holistically & individually



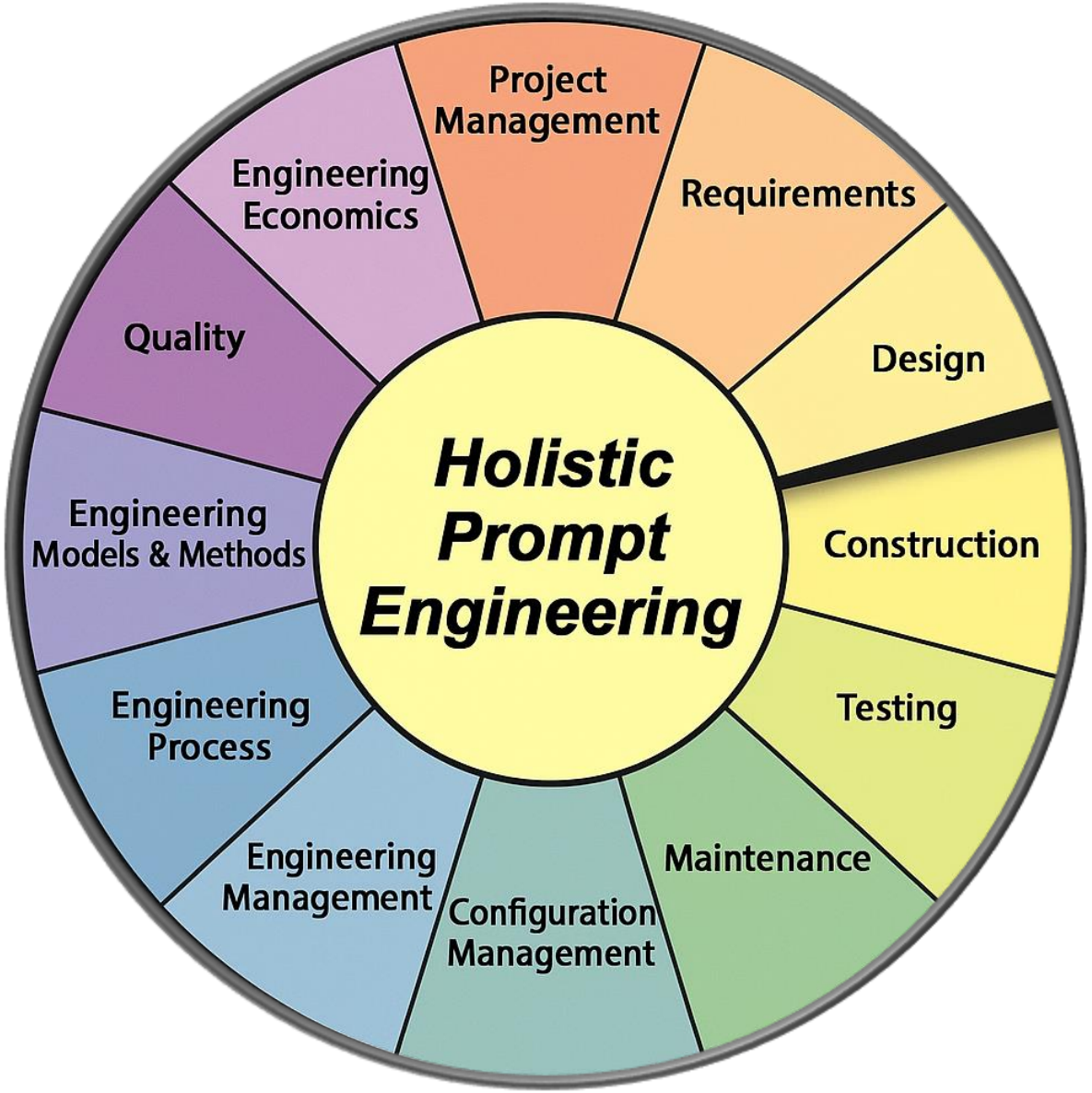
What We Can Do to Prepare Our Students (& Ourselves) for AI

- Prepare for the near future where programmers are outnumbered by "computational thinkers"



What We Can Do to Prepare Our Students (& Ourselves) for AI

- Prepare for the near future where programmers are outnumbered by "computational thinkers"
  - Requires new approaches to prompt engineering, technical debt, etc.



See devops.com/will-the-rise-of-generative-ai-increase-technical-debt

Remember the Lesson of King Canute!

See en.wikipedia.org/wiki/King_Canute_and_the_tide

End of the Coming Commoditization of Computational Thinking & Its Impact on Computer & Data Science