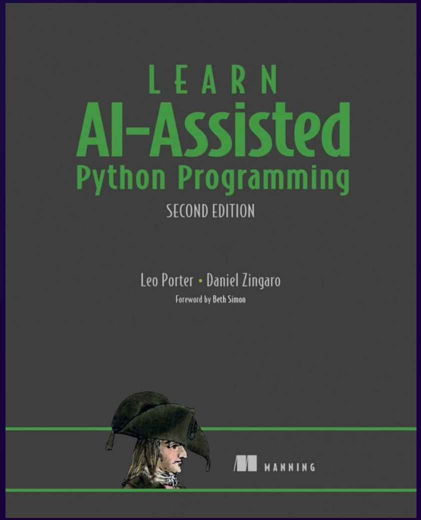
Integrating Generative AI with Introductory Programming

Leo Porter	UC San Diego
Dan Zingaro	University of Toronto



LEARN
AI-Assisted
Python Programming
SECOND EDITION

Leo Porter • Daniel Zingaro
Foreword by Beth Simon

MANNING

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Computer Science and Engineering

UNIVERSITY OF TORONTO

Key Takeaways

1. The research is moving quickly and has started to focus on uncovering how to best help students learn with an LLM

2. Learning to write software with an LLM requires teaching a different mix of skills than in the past

3. Our CS1-LLM pilot has encouraging findings with regard to student perceptions and the ability for students to create large software projects

Our Plan Today

1. **Brief summary of current research**
2. Goals for students learning in CS1-LLM
3. Our CS1-LLM course pilot
4. Initial findings from our CS1-LLM course
5. Discussion and Q&A

LLMs solve CS1 Assessments

- LLMs solving CS1 assignments [1]
  - Copilot solved 47.6% of problems on its first attempt and that went up to 79.5% after prompt engineering
- LLM solving on CS1 exams [2]
  - In 2021, Codex received 78.5% on Exam 1 and 78% on Exam 2
  - In 2023, GPT-4 received 99.5% on Exam 1 and 94.4% on Exam 2

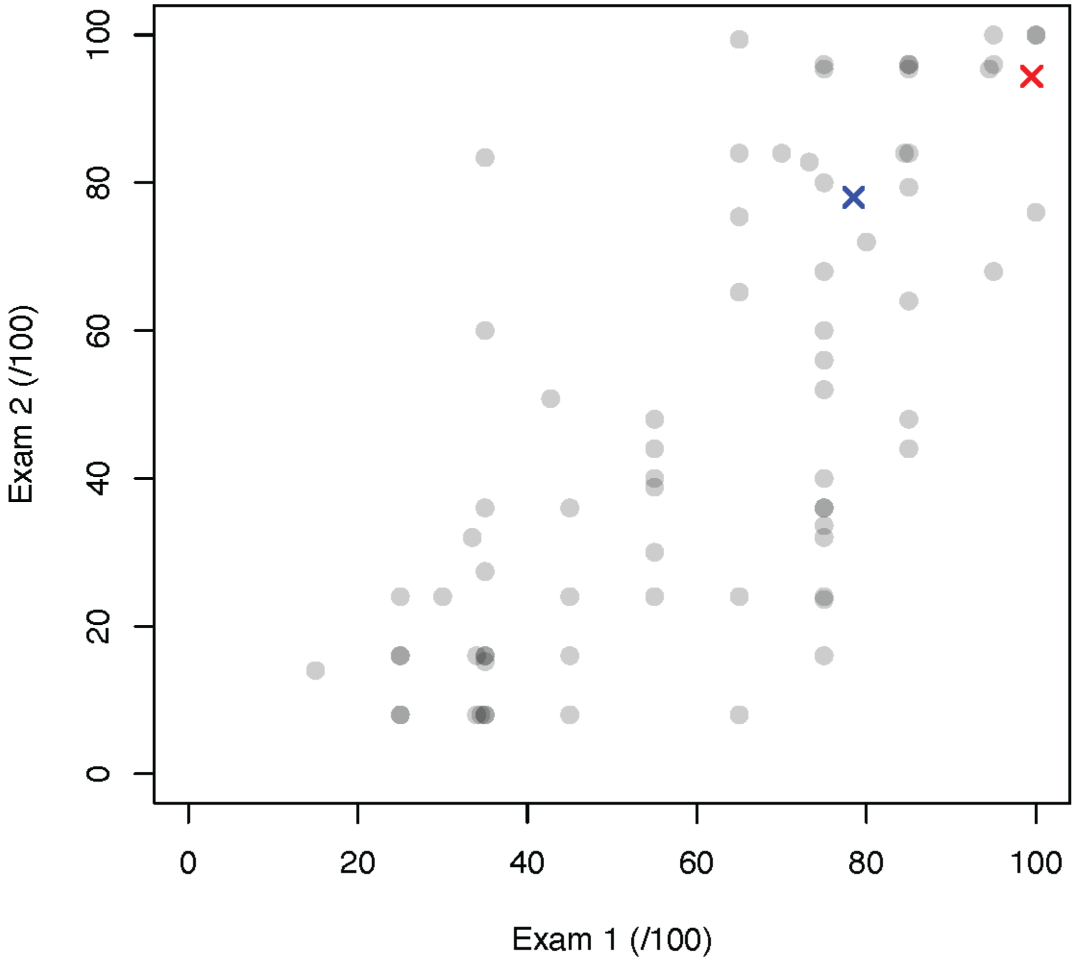


Figure from [2]. Student + AI performance on CS1 exams. Blue is Codex, Red is GPT-4

1. Denny et al. Conversing with Copilot: Exploring Prompt Engineering for Solving CS1 Problems Using Natural Language. ACM SIGCSE 2023.
2. Denny et al. Computing Education in the Era of Generative AI. CACM 2024.

GenAI can be an effective tutor

Tools include: CodeHelp [1], CodeAid [2], etc.

- Often include guardrails to limit responses

Students value these tools [3,4]

- Encouraging tone
- Ease of access
- Focus on learning
- Conversing in their native language

Particularly in comparison with human tutors

- Human tutors often give away answers and pass judgements [5]

1. Liffiton et al. 2024. CodeHelp: Using Large Language Models with Guardrails for Scalable Support in Programming Classes. Koli Calling '23.
2. Kazemitabaar et al. CodeAid: Evaluating a Classroom Deployment of an LLM-based Programming Assistant that Balances Student and Educator Needs. CHI 2024.
3. Denny et al. Desirable Characteristics for AI Teaching Assistants in Programming Education. ITiCSE 2024.
4. Villegas Molina et al. Leveraging LLM Tutoring Systems for Non-Native English Speakers in Introductory CS Courses. arXiv: 2411.02725
5. Krause-Levy et al. An exploration of student-tutor interactions in computing. ITiCSE 2022.

Image Credit @hansakuluarachc

Evidence is mixed about learning to program

- Novices learned to program better when using GenAI [1]
  - Learners performed a series of tasks generating and modifying code
  - Learners with GenAI could write code and modify code better, even when GenAI was taken away
- Novices may struggle to communicate with GenAI [2]
  - Prompts missing omitting important details are less effective
  - Novices need to learn what details are important to convey
- Students encounter new metacognitive challenges using GenAI [3]
  - Students are more effective at programming tasks with GenAI, but encounter different kinds of challenges working with GenAI
  - Despite being taught about learning to program with GenAI, interviewers had concerns that students may not be able to code well if the tool were taken away

1. Kazemitabaar et al. Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. CHI 2023.
2. Lucchetti et al. Substance Beats Style: Why Beginning Students Fail to Code with LLMs. https://arxiv.org/abs/2410.19792
3. Prather et al. The Widening Gap: The Benefits and Harms of Generative AI for Novice Programmers. ICER 2024.

Engagement with AI Generated Code

- The research is quickly moving from "should we teach with GenAI?" to "HOW should we teach with GenAI?"
  - We likely need to find ways to slow students down when using GenAI
  - This "friction" has to engage students in meaningful activities
    - e.g. "Lead-and-Reveal": before showing each line of code, the AI asks the student a question that they must answer correctly [1]
    - e.g. The GenAI should help users reflect on the decisions they’re making [2]
  - These techniques help students engage critically, rather than passively accepting what the AI gives them

1. Kazemitabaar et al. Exploring the Design Space of Cognitive Engagement Techniques with AI-Generated Code for Enhanced Learning. arXiv:2410.08922.
2. Tankelevitch et al. 2024. The Metacognitive Demands and Opportunities of Generative AI. CHI '24.

Educators views are changing

- Concerns over assessments
  - Educators are decreasing emphasis on classic take-home assignments
  - Switching to invigilated exams, oral exams, or assessing process

- Skills are changing
  - **75%** of educators acknowledge that the skills to program are changing because of GenAI
  - **30%** of educators are integrating it into their classes

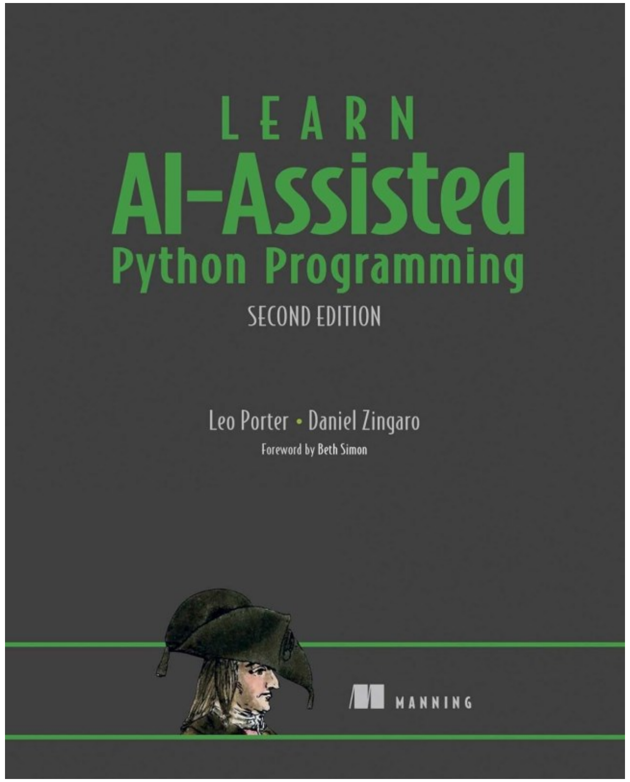
Prather et al. How Are Instructors Incorporating Generative AI into Teaching Computing? ITiCSE 2024 Working Group.

Our Plan Today

1. Brief summary of current research
2. **Goals for students learning using our book**
3. Our CS1-LLM course pilot
4. Initial findings from our CS1-LLM course
5. Discussion and Q&A

A New Approach to Learning Programming

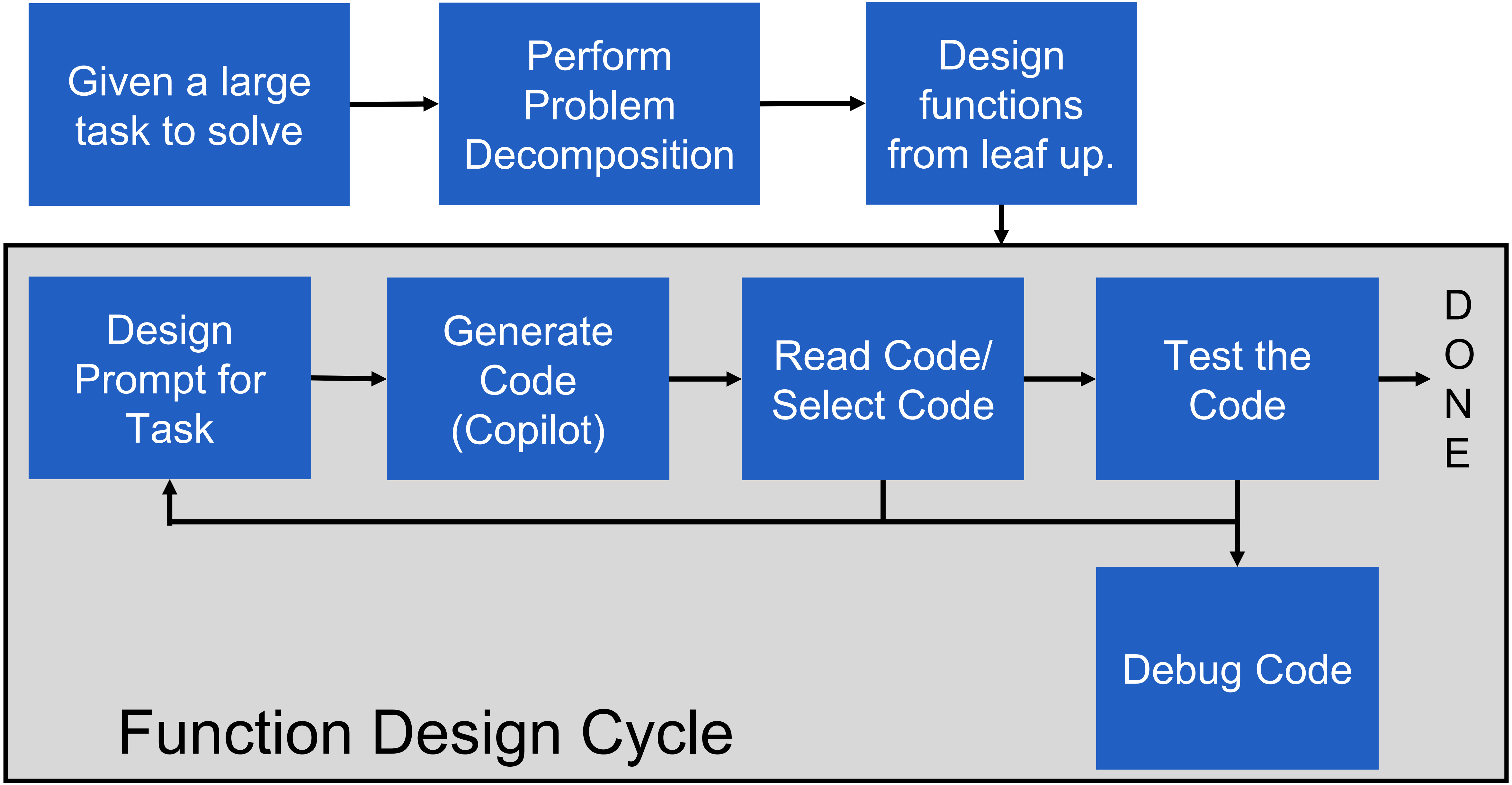
- Key Goals:
  - Learn key skills essential to successful software design with LLMs



E-mail for instructor copy:
leporter@ucsd.edu

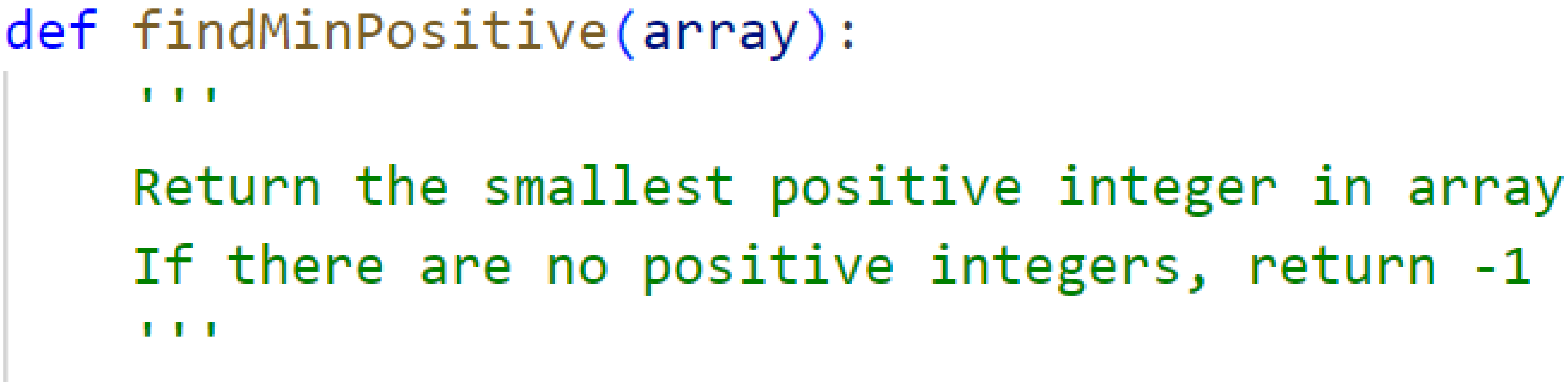
Leo Porter and Daniel Zingaro. Learn AI-Assisted Python Programming: With GitHub Copilot and ChatGPT. Second Edition. Manning Publications. 2024.

The workflow when writing software with Copilot



Skill: Function Design and Prompting

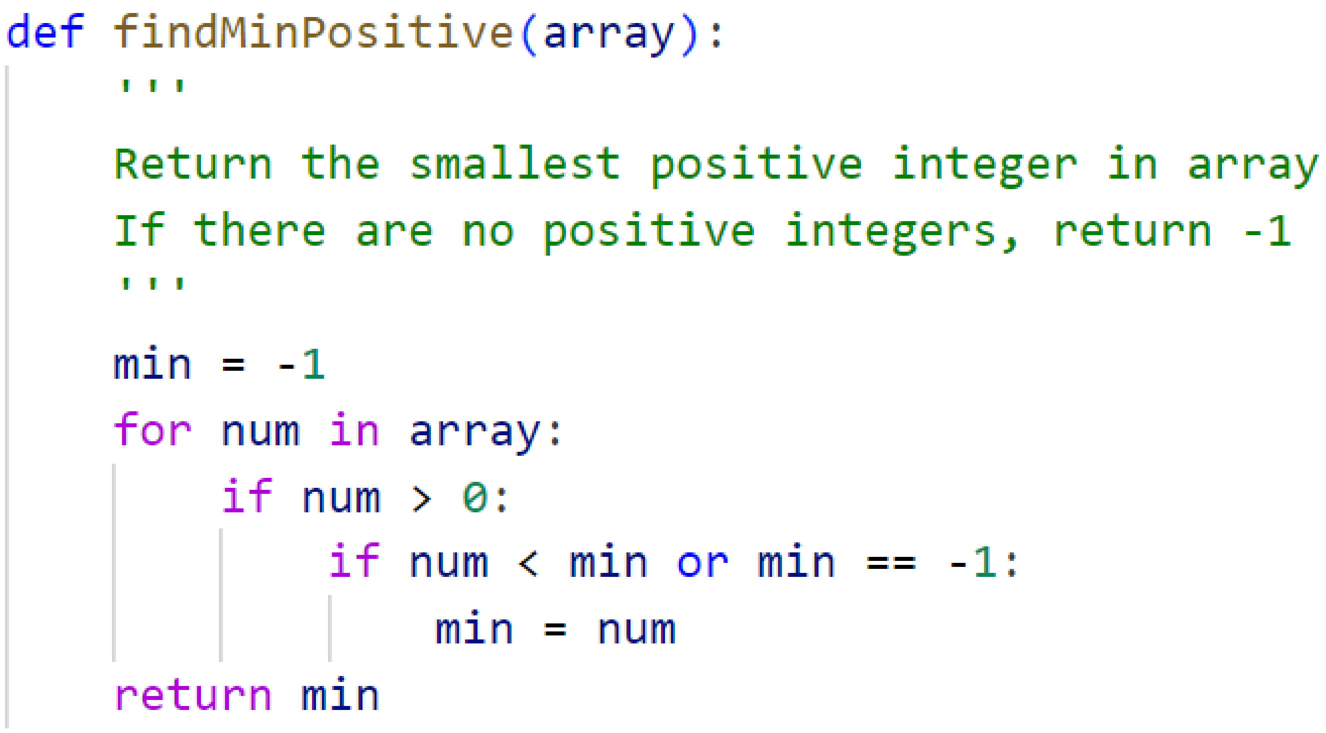
- Create the function signature
- Describe the behavior of the function
  - In sufficient detail for Copilot to try to solve it



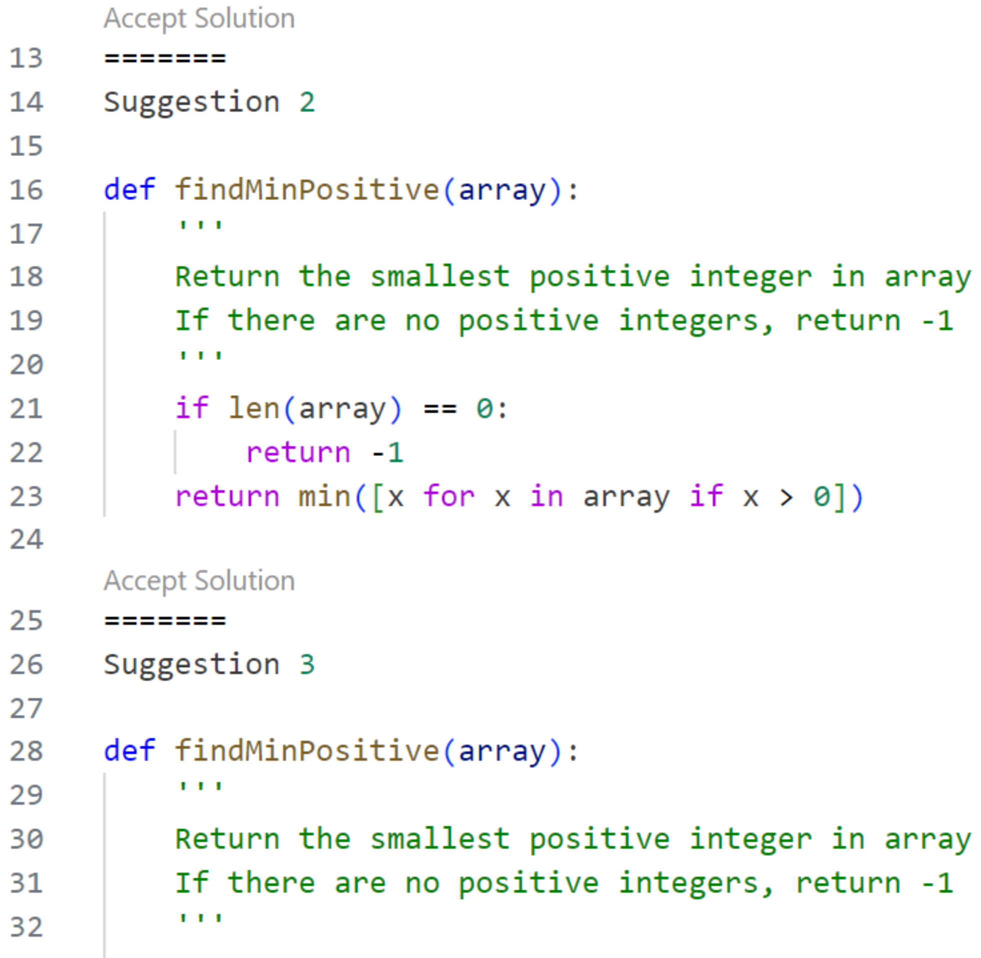
```
def findMinPositive(array):
    '''
    Return the smallest positive integer in array
    If there are no positive integers, return -1
    '''
```

Skill: Code Reading and Selection

- Be able to understand code produced by the LLM
- Be able to read multiple suggestions of code and select one that seems likely to help solve the problem



```python
def findMinPositive(array):
    '''
    Return the smallest positive integer in array
    If there are no positive integers, return -1
    '''
    min = -1
    for num in array:
        if num > 0:
            if num < min or min == -1:
                min = num
    return min
```



Accept Solution

```
=======
Suggestion 2

def findMinPositive(array):
    '''
    Return the smallest positive integer in array
    If there are no positive integers, return -1
    '''
    if len(array) == 0:
        return -1
    return min([x for x in array if x > 0])
```

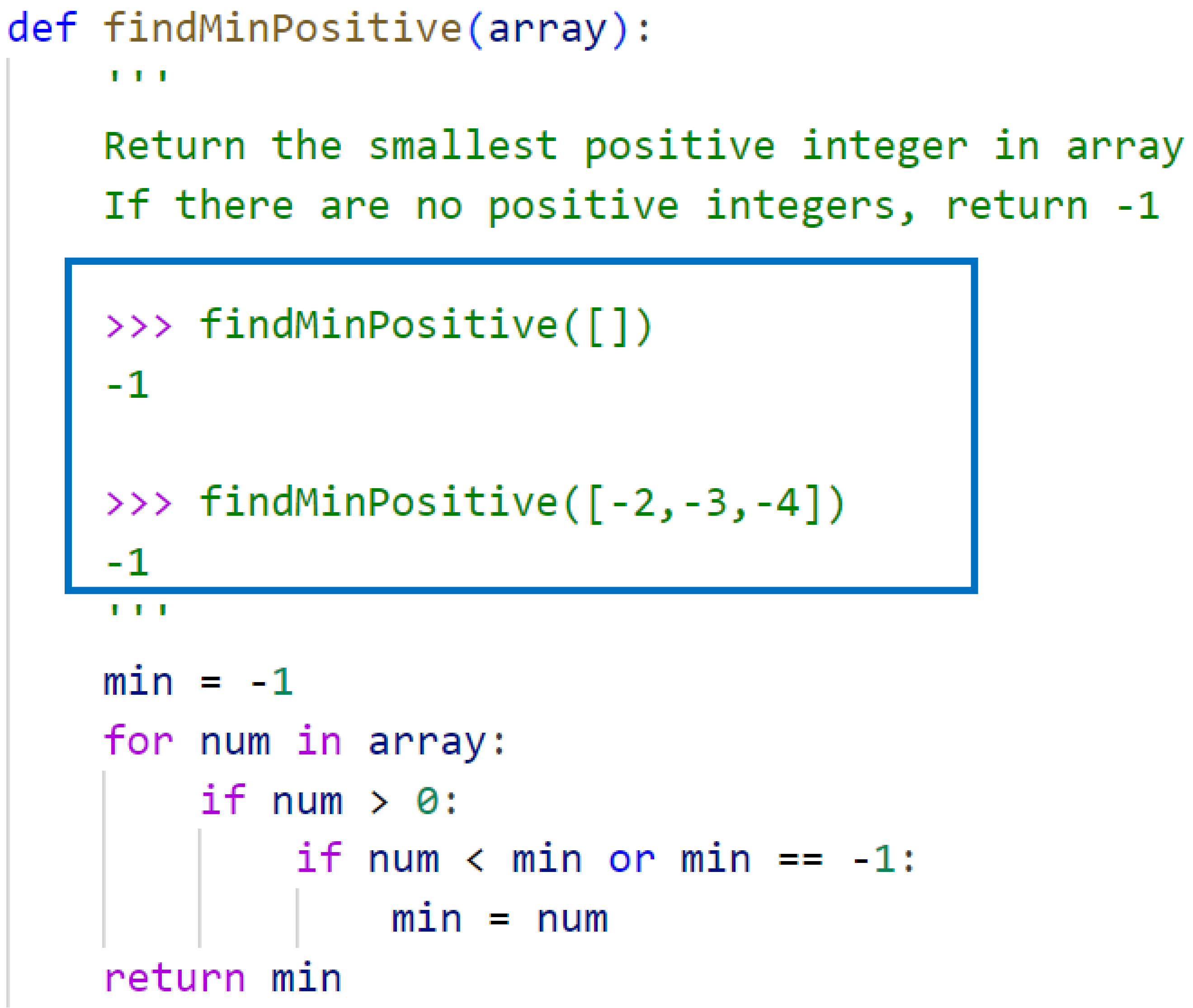
Accept Solution

```
=======
Suggestion 3

def findMinPositive(array):
    '''
    Return the smallest positive integer in array
    If there are no positive integers, return -1
    '''
```

Skill: Code Testing

- Open- and Closed-box testing
- Understand common case vs. edge-cases
  - understand code well enough to identify possible edge cases
- Use doctest to test the code



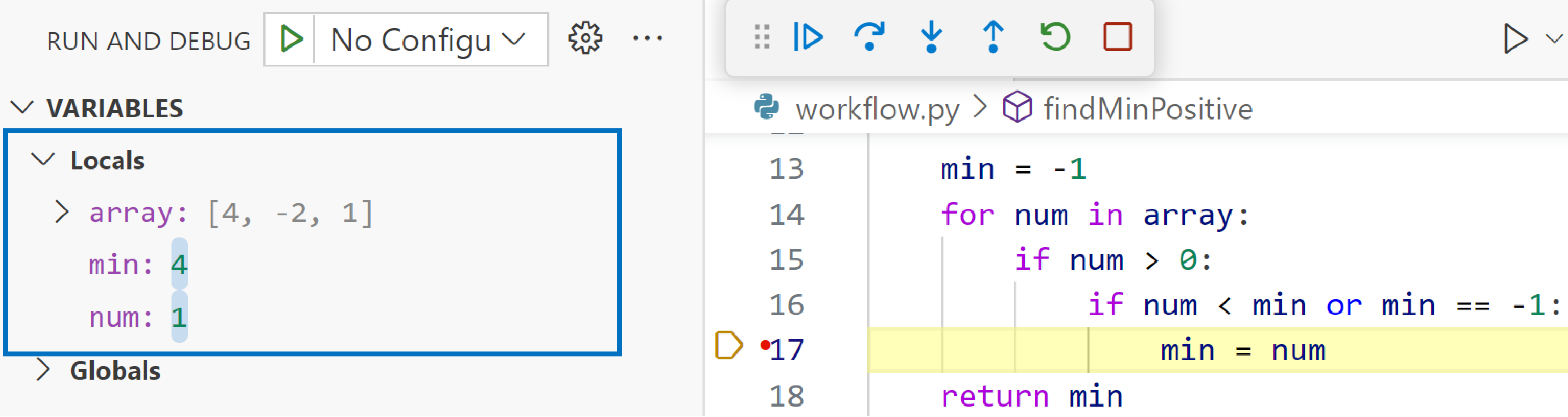
```
def findMinPositive(array):
    '''
    Return the smallest positive integer in array
    If there are no positive integers, return -1

    >>> findMinPositive([])
    -1

    >>> findMinPositive([-2,-3,-4])
    -1
    '''
    min = -1
    for num in array:
        if num > 0:
            if num < min or min == -1:
                min = num
    return min
```

Skill: Debugging

- Use VSCode Debugger to gain insight into the behavior of the program
  - Know how to add breakpoints, follow variable state during execution
- Use it to gain insight into how to modify the code



Locals

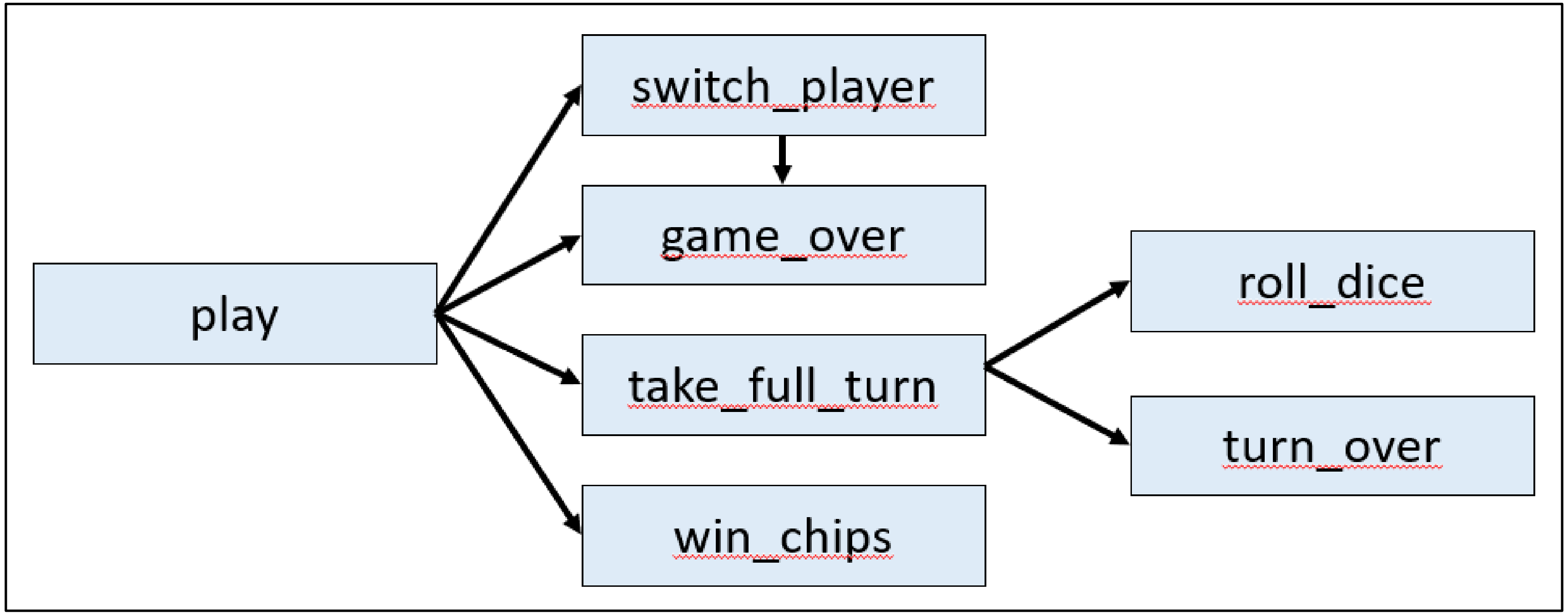
- array: [4, -2, 1]
- min: 4
- num: 1

Globals

```
    min = -1
    for num in array:
        if num > 0:
            if num < min or min == -1:
                min = num
    return min
```

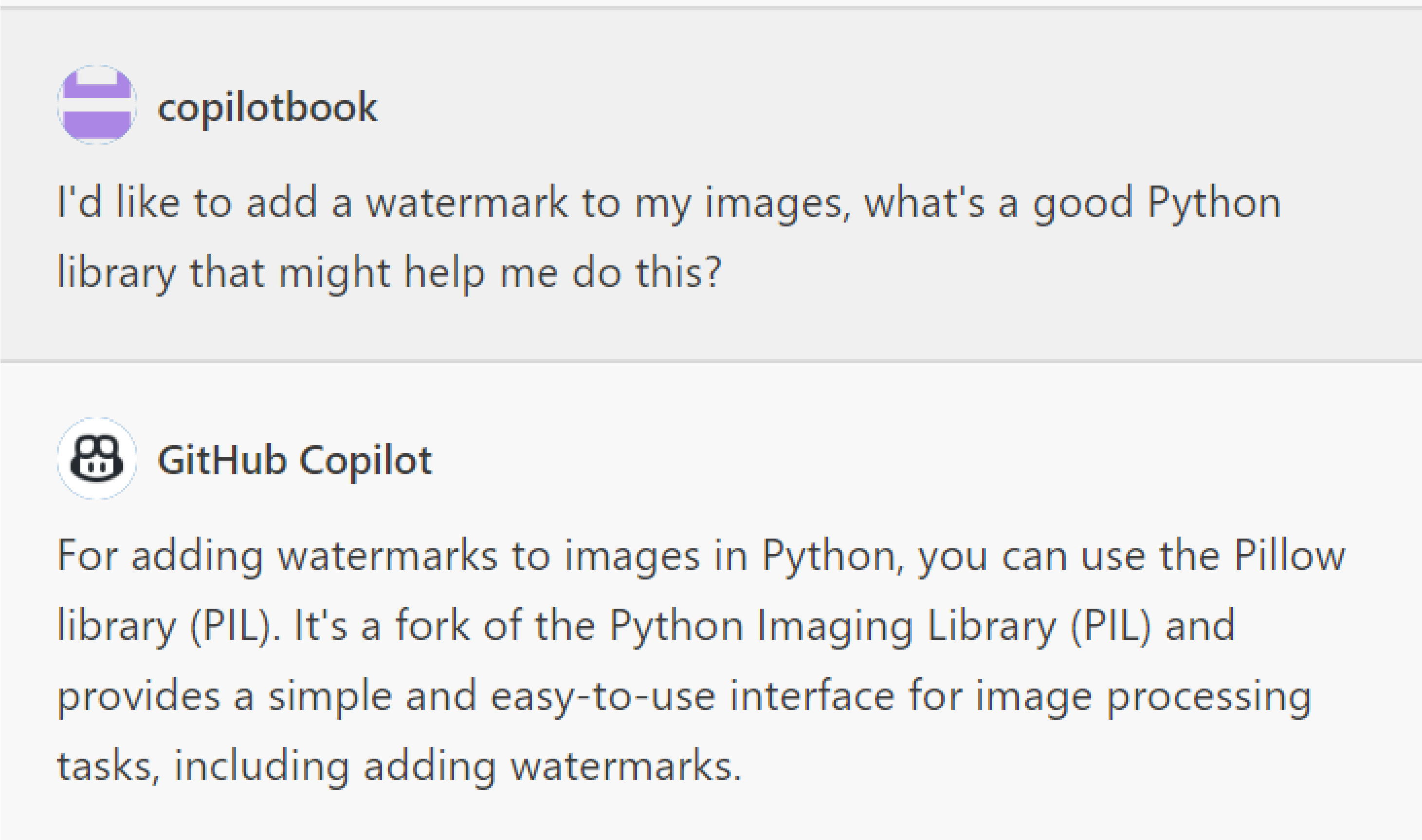
Skill: Problem Decomposition

- Necessary to solve any medium/large task and we want students solving real tasks
  - Need to be able to break this into functions
- Often helpful for debugging
  - Breaking a problem into smaller pieces helps Copilot generate correct code



Skill: Identifying Useful Modules

- Be able to explore new programming domains and leverage existing libraries
- Conversations with Copilot Chat can make it easy to identify useful libraries



copilotbook

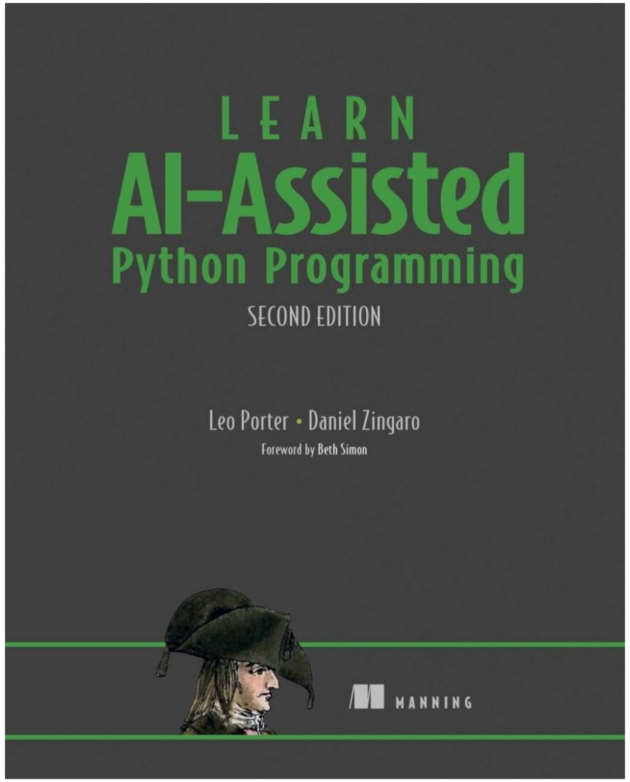
I'd like to add a watermark to my images, what's a good Python library that might help me do this?

GitHub Copilot

For adding watermarks to images in Python, you can use the Pillow library (PIL). It's a fork of the Python Imaging Library (PIL) and provides a simple and easy-to-use interface for image processing tasks, including adding watermarks.

A New Approach to Learning Programming

- Key Goals:
  - Learn key skills essential to successful software design with LLMs
    - Code reading
    - Testing
    - Debugging
    - Problem decomposition
  - Use LLM features to help them understand code
  - Have readers aware of the ethical/legal challenges around LLMs
  - Have students interact with LLMs to solve problems that are meaningful to them



E-mail for instructor copy:
leporter@ucsd.edu

Leo Porter and Daniel Zingaro. Learn AI-Assisted Python Programming: With GitHub Copilot and ChatGPT. Second Edition. Manning Publications. 2024.

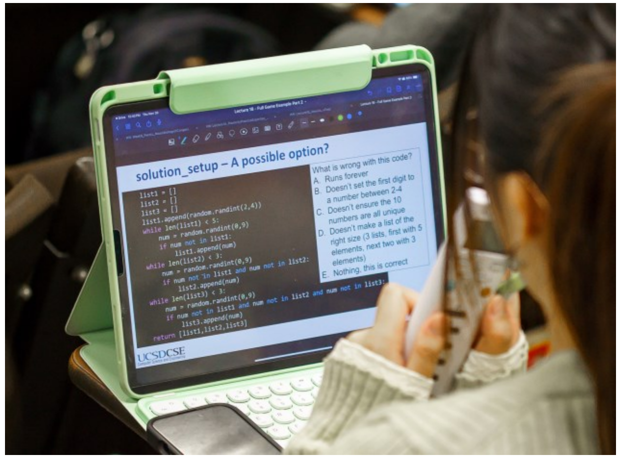
Our Plan Today

1. Brief summary of current research
2. Goals for students learning using our book
3. **Our CS1-LLM course pilot**
4. Initial findings from our CS1-LLM course
5. Discussion and Q&A

CS1-LLM Course at UCSD

- Created new learning goals for the course
- Used GitHub Copilot as the LLM for the course
- Discussed ethical implications of GenAI
- Had large software projects in multiple domains
    - Data Science, Image Manipulation, and Games

Credit: Ismael and Patringenaru



solution_setup – A possible option?

```
list1 = []
list2 = []
list3 = []
list1.append(random.randint(2,4))
while len(list1) < 5:
    num = random.randint(0,9)
    if num not in list1:
        list1.append(num)
while len(list2) < 3:
    num = random.randint(0,9)
    if num not in list1 and num not in list2:
        list2.append(num)
while len(list3) < 3:
    num = random.randint(0,9)
    if num not in list1 and num not in list2 and num not in list3:
        list3.append(num)
return [list1,list2,list3]
```

What is wrong with this code?

A. Runs forever
B. Doesn't set the first digit to a number between 2-4
C. Doesn't ensure the 10 numbers are all unique
D. Doesn't make a list of the right size (3 lists, first with 5 elements, next two with 3 elements)
E. Nothing, this is correct

Credit: Ismael and Patringenaru

CS1-LLM – Course Schedule

Week	Topics(s)	
1	Functions and Working with Copilot	Code Reading / Workflow with Copilot
2	Variables, Conditionals, Memory Models	
3	Loops, Strings, Testing, VSCode Debugger	
4	Loops, Lists, Files, Problem Decomposition	
5	Intro to Data Science, Dictionaries	Data Science
6	Revisit Problem Decomposition and Testing	
7	Intro to Images, PIL, Image Filters	Images
8	Copying Images, Intro to Games and Randomness	
9	Large Game Example	Game Design
10	Python Modules and Automating Tedious Tasks	

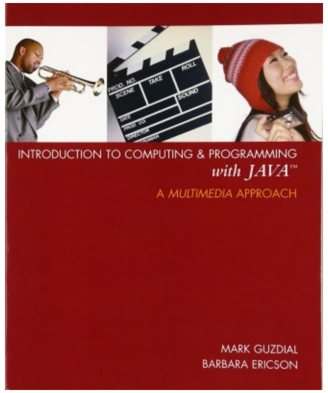
CS1-LLM – Assessments

Assessment	Percentage of Grade	Details
Clickers	5%	Peer Instruction
Reading Quizzes	5%	Pre-class Preparation
Homework	15%	PrairieLearn - with/without Copilot
Labs	10%	PrairieLearn - Small Programming Activities
Projects	10%	**Open-ended Programming Projects with Copilot**
Quizzes	30%	PrairieLearn - Code Reading, Testing, Parsons (**w/o Copilot**)
Final Exam	25%	PrairieLearn - Part 1: Code Reading/Parsons; Part 2: Code Writing **without Copilot**; Part 3: Code Writing **with Copilot**

CS1-LLM – Project Features

- Three of the domains taught in the course
  - Data Science, Images, Games

- Open-ended for creativity
  - Project 1: Pick a dataset on Kaggle, ask an interesting question of it, decompose the problem, use Copilot to help write the code, test it.
  - Project 2: Using free access images or your own images, create an image collage that includes multiple filters applied using PIL [1]
  - Project 3: Design a text-based game or game simulator



Guzdial and Ericson. Introduction to Computing and Programming with Java. A Multimedia Approach. Pearson. 2006.

CS1-LLM – Project Grading

- Graded via Code and Video Submission
  - (1 min) What was the goal of the project
  - (1 min) Talk through the problem decomposition
  - (3 min) Walk through one function and explain how it works

Our Plan Today

1. Brief summary of current research
2. Goals for students learning using our book
3. Our CS1-LLM course pilot
4. **Initial findings from our CS1-LLM course**
5. Discussion and Q&A

CS1-LLM Pilot Course at UCSD in Fall 2023

- Approximately 550 students enrolled in the course
  - Roughly 275 students in each of 2 sections
- Staff: 6 graduate TAs, 33 undergraduate tutors
- Course demographics:
  - Majority female (52%), non-CS majors (66%)
  - BLNPI (33%), First-Generation (43%), Pell eligible (47%)

Vadaparty, A., Zingaro, D., Smith, D., Padala, M., Alvarado, C., Benario, J., Porter, L. Experience Report: Integrating LLMs into CS1 Instruction. ITiCSE 2024. **bit.ly/CS1-LLMs**

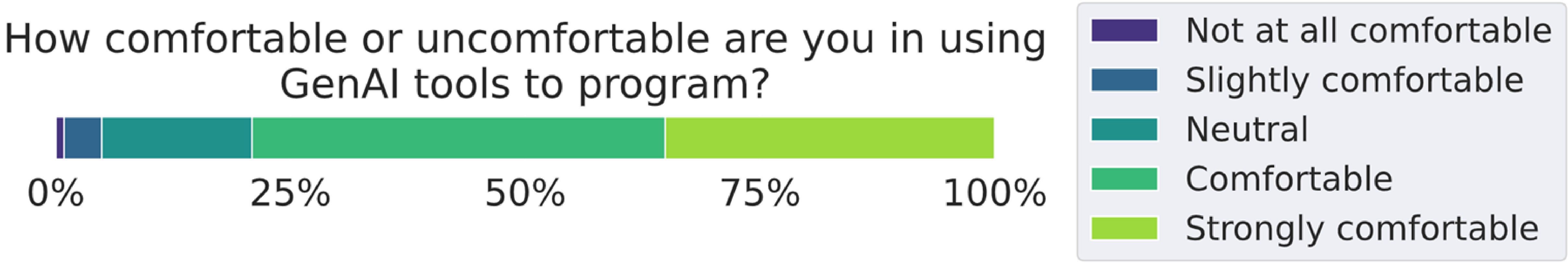
CS1-LLM – Programming Project Outcomes

- Non-CS majors often brought in ideas from their majors
  - Art majors incorporating their own art
  - Neuroscience majors performing data analysis on stroke data
- Impressive submissions
  - Graders asked for ways to give bonus points to the many students who went above and beyond. We ultimately limited how many the graders could award.
  - Students weren’t aware that this is beyond the scope of a standard CS1

Student Perceptions: Surveys

- Three Surveys: Pre, Mid, and End-quarter surveys
  - Weeks 1, 7, and 10
  - Number of responses (out of 552 students): 473, 400, 315

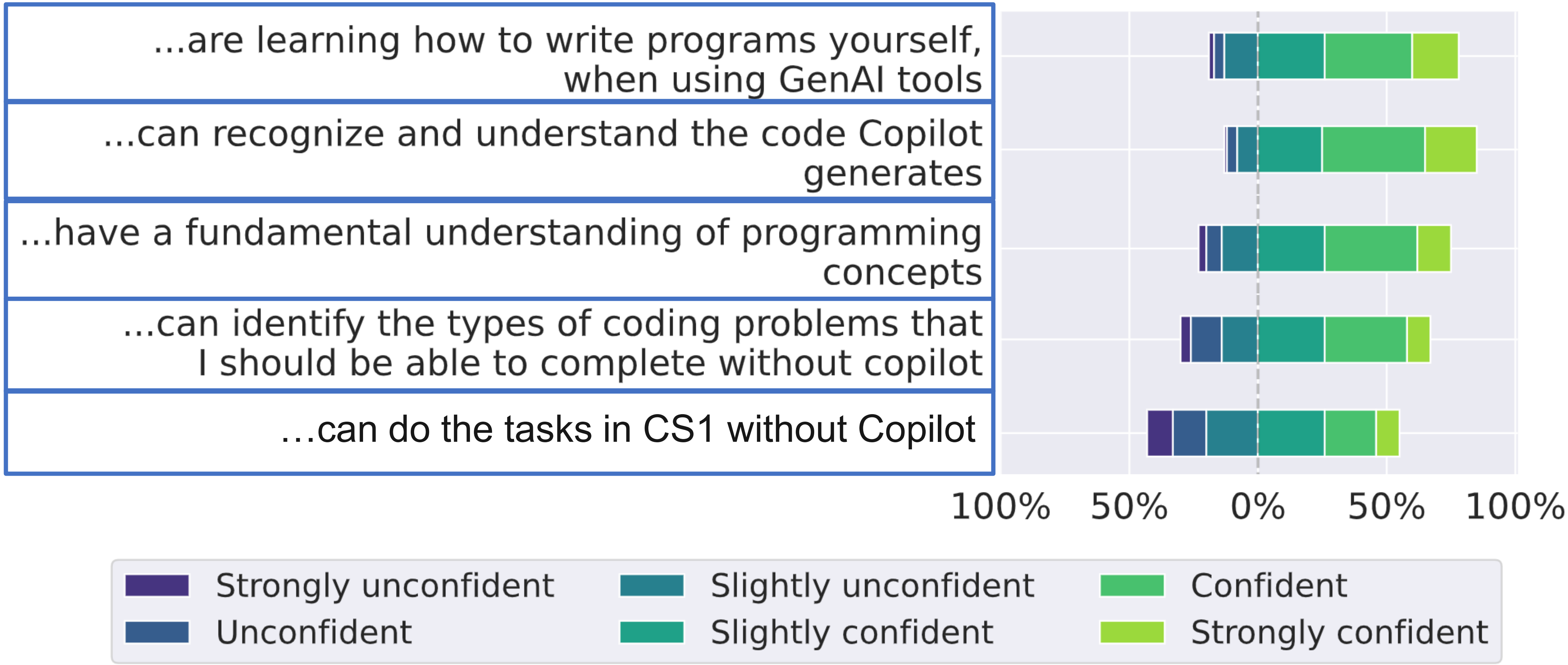
Students are Comfortable Using GenAI



How comfortable or uncomfortable are you in using GenAI tools to program?

Student Confidence related to GenAI

How confident or unconfident are you that you



Mixed feelings about Copilot

Students were asked: “How did you feel about working with Copilot as you learned to program this quarter?”

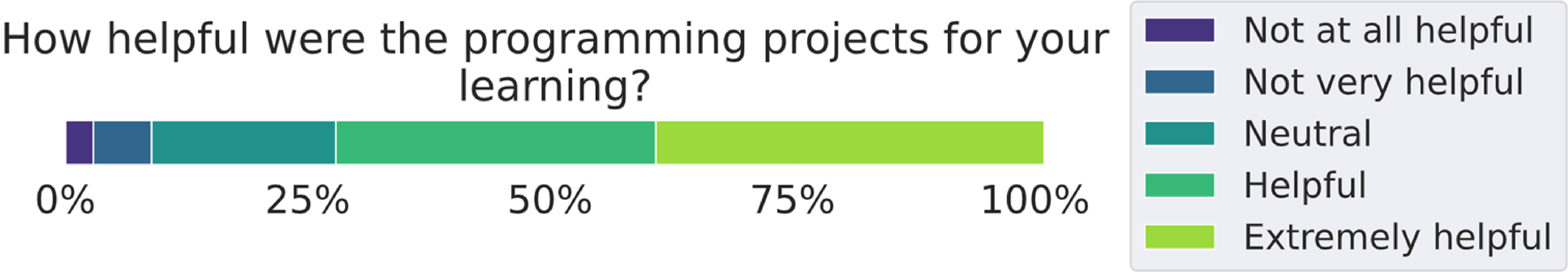
- Appreciate its value

  “I feel working with Copilot was helpful since you could get basic code ***quicker*** *and ask any* ***questions specifically about your code****.”*

- Appreciate Copilot, but perceptions of over-reliance:

  “I think Copilot is a ***helpful tool****, but using it so often has made me* ***dependent*** *on it. If I had to code on the spot without Copilot I* ***wouldn't be able to do it****.”*

Students Valued the Projects for their Learning



74% of students felt it was helpful/extremely helpful

Student Perceptions of the Projects

- Largely positive responses; creative freedom was appreciated
  “Programming projects were fun and the amount of ***freedom*** *that was given added to that. The projects also helped me understand how to put* ***everything that we have learned*** *so far into a project that I could be proud of.”*

Student Perceptions of the Projects

- Largely positive responses; creative freedom was appreciated

  "Programming projects were fun and the amount of ***freedom*** *that was given added to that. The projects also helped me understand how to put* ***everything that we have learned*** *so far into a project that I could be proud of."*

- Sample projects and/or clearer guidelines wanted

  "Fun and interesting, but maybe they had a little too much freedom? Maybe a few more ***specific guidelines*** *or suggestions on how to get started."*

Student Motivations Impact Computing Interest

- Measured Student Motivation as Achievement Goals:

Goal	Example Item
Mastery	*My goal is to learn as much as possible.*
Normative	*My aim is to perform well relative to other students.*
Appearance	*One of my goals is to show others that I'm good at my class work.*

- At the end of the course, measured student interest in computing: e.g.: *I think what we are learning in this class is interesting.*
- In traditional CS1 courses, Mastery goals predict interest [1]
- In our CS1-LLM course, Mastery goals *still* predict interest [2]

1. Zingaro and Porter. Impact of Student Achievement Goals on CS1 Outcomes. SIGCSE 2016.
2. Vadaparty, Geng, Smith, Benario, Zingaro, Porter. Achievement Goals in CS1-LLM. ACE 2025. https://bit.ly/CS1LLM-ACE25

Takeaways from our CS1-LLM course

- **Projects empowered students to be creative and engage in areas of their interest**
  - If non-majors only take one CS course, we'd want it to be this one.

- Take careful consideration of aligning assessments to learning goals

- Updating the course can be challenging – we've shared our course materials for interested instructors

- This was just the pilot – we look forward to continued improvements

Our Plan Today

1. What we know from the growing body of research
2. Design Goals for integrating LLMs into CS1
3. Activity – How to assess students in the presence of LLMs
4. Our findings from our CS1-LLM course
5. **Discussion and Q&A**

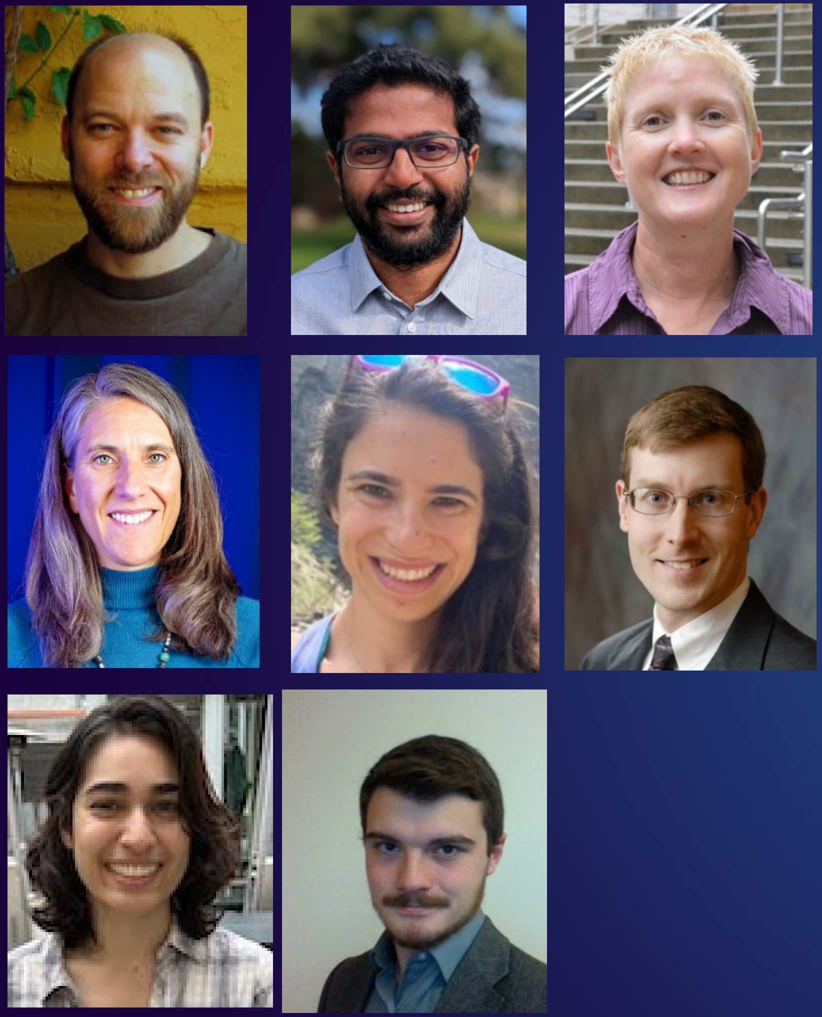
Partnering with other Committed Educators

Signup for Google's 5-course Sequence
https://bit.ly/FacultyGenAI

- Over 350 instructors have asked for book copies
  - Providing support for their classes where we can
  - Working on releasing more materials for instructors
- With NSF support, revising this CS1-LLM course
- Partnering with teams at Google
  - Research partnership with Education for Social Impact team
  - Outreach partnership with TechExchange
  - TechExchange has developed 5 CS courses that incorporate GenAI
    - CS1, CS2, Algorithms, Software Engineering, Project Management

Google's Summit on "Learning in the AI Era"

Thank you and questions!



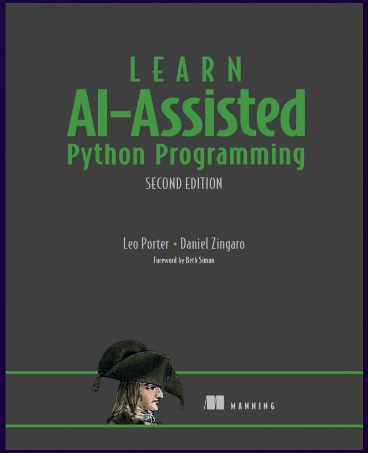
E-mail for instructor book copy: leporter@ucsd.edu
Link to second edition: bit.ly/CS1-LLM-Book

Link to Experience Report: bit.ly/CS1-LLMs

Link to Achievement Goals Paper:
https://bit.ly/CS1LLM-ACE25

Link to Course Materials:
https://github.com/copilotbook/CS1-LLM

Signup for Google's 5-course Sequence
https://bit.ly/FacultyGenAI



LEARN
AI-Assisted
Python Programming
SECOND EDITION

Leo Porter • Daniel Zingaro
Foreword by Beth Simon

MANNING

Discussion Questions

- For instructors, what support would you need to transition toward a CS1-LLM course?
- How can we work together to determine the new learning goals for CS1-LLM?
- How can we help students learn the skills we care about?
- Is it fair to assess students on their ability to write code from scratch (recognizing LLMs can perform this task for them)?
- If instructors want students to use an LLM with guardrails, how can they promote those tools over students just turning to ChatGPT?