Using Generative AI to Create Personalised Parsons Problems and Explanations:

http://tinyurl.com/ericsonJan2024

Dr. Barbara (Barb) Ericson
Assistant Professor
School of Information
University of Michigan
barbarer@umich.edu

Xinying Hou
PhD Student
School of Information
University of Michigan
xyhou@umich.edu

Barb's History

- BS in CS from Wayne State Un in 83
- Was a software engineer
  - General Motors Research Labs
  - Bell Communications Research
- MS in CS and Eng from Un of Michigan in 86
- Director of Outreach at Georgia Tech (GT)
- PhD from GT in 2018 – HCC

Research Areas

- Active learning
  - Interactive ebooks
  - Parsons problems
  - Peer Instruction
  - POGIL
- Social justice
- Near-peer mentoring programs
- Curriculum development
  - Media Computation
  - MOOC

Sisters Rise Up

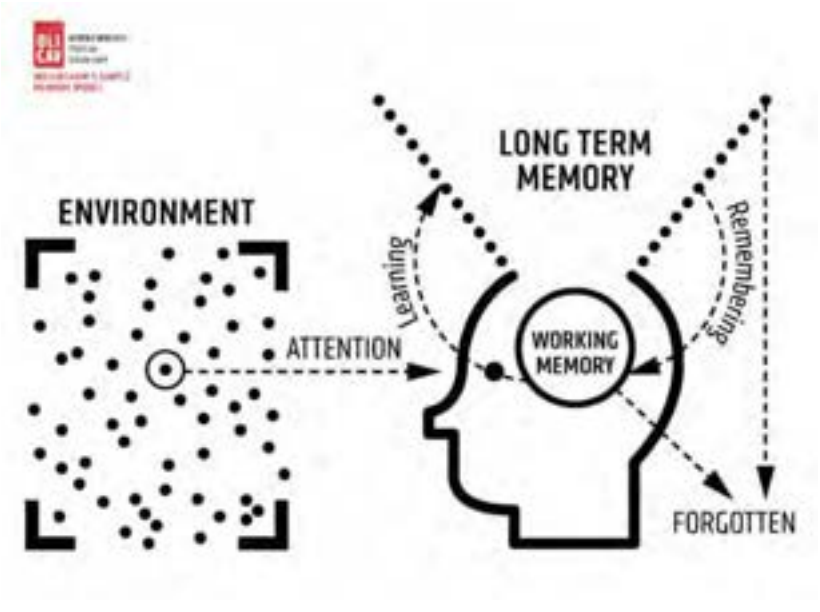
Active Learning

- ICAP theory of Cognitive Engagement
  - Interactive > constructive > active > passive
- Interactive – peer discussion
- Constructive – creating something new
- Active – moving something
- Passive – typical lecture

Chi, Michelene and Wylie, Ruth 2014

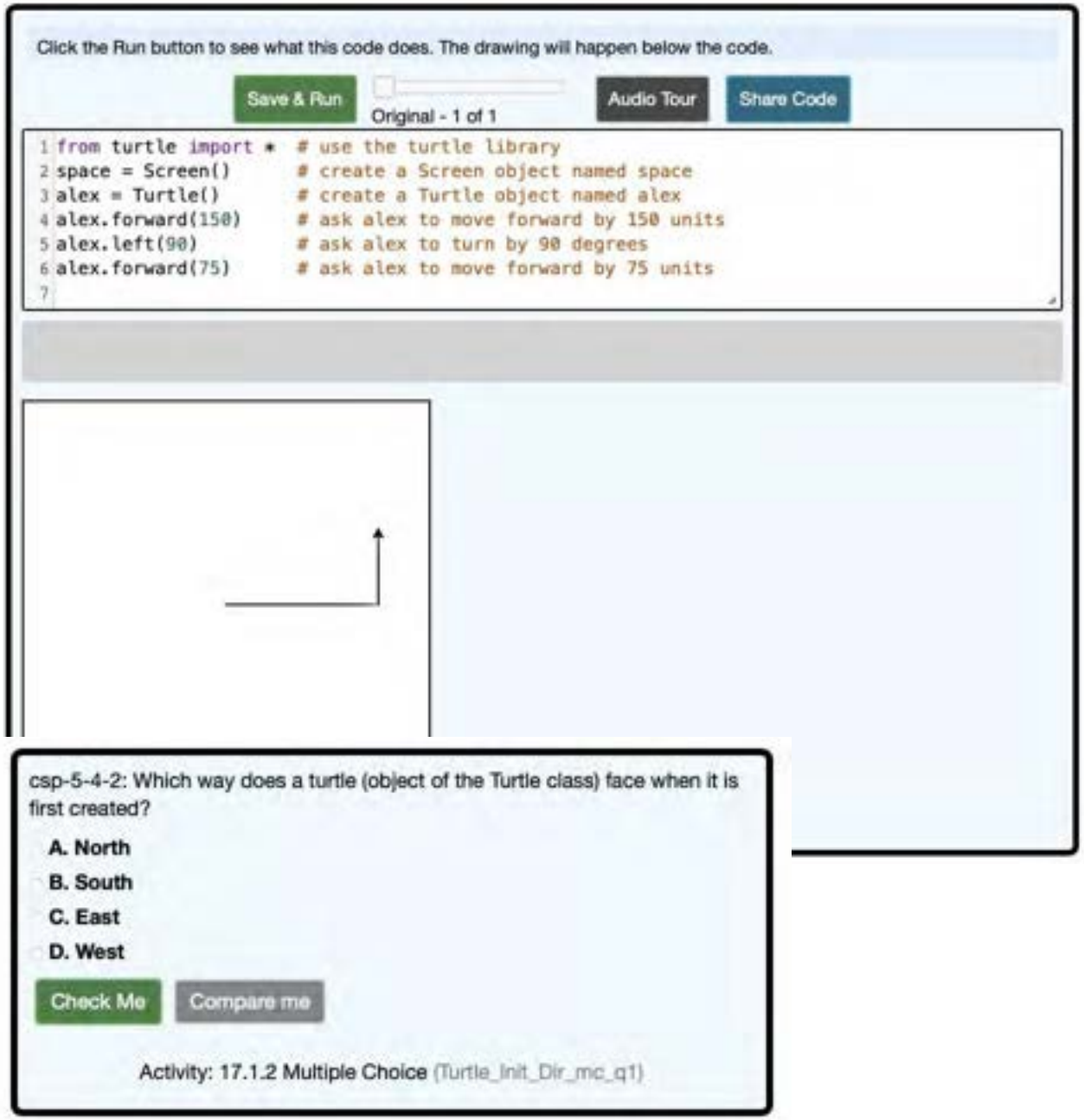
Cognitive Load Theory

- Developed by *John Sweller* in the 1980s
- Reduce cognitive load to free up working memory to improve learning
  - Completion task better than whole task - Van Merriënboer and De Croock. 1992.
  - Cognitive load is based on prior knowledge
- Predicts effects: worked example effect
  - Examples with interleaved practice best - Trafton & Reiser, 1993



Interactive Ebooks

- Educational psychology principles
  - Worked examples plus practice with immediate feedback
  - Timed exams
  - Spaced practice tool
  - Lower cognitive load practice problems – Parsons
- My ebooks
  - CS Awesome - over 30,000 reg users
  - Python for Everybody - Interactive



Runestone Academy

- Founder
  - *Democratizing textbooks for the 21st century*
  - Started with one ebook in 2011
- 30+ free ebooks for computing and math
  - Over
- https://tinyurl.com/bdz2fhzy

Brad Miller

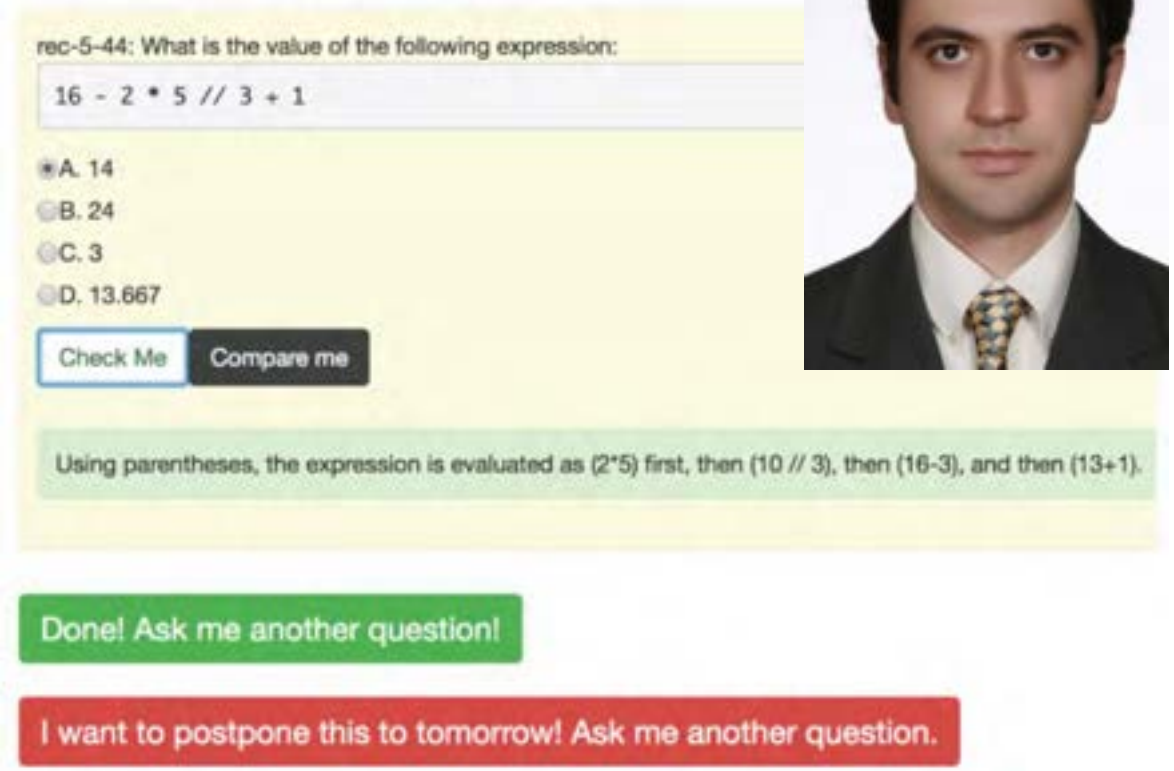
Timed Exams

- Instructors can create auto graded timed exams
- Can have MCQs, fill in the blank, mixed-up code (Parsons) problems, and write code problems with unit tests
  - Can compile code
- Students don’t get feedback on their answers during the exam

Spaced-Practice Tool

- Spaced practice
  - Displays previous questions from an interactive ebook
    - Based on user's performance
  - Algorithm predicts when likely to forget a topic
  - Can earn points for answering X questions correctly over Y days
- Each hour of using the practice tool was associated with > 1% increase in the final exam grade

Iman Yeckehzaare



YeckehZaare, Resnick, Ericson, 2019

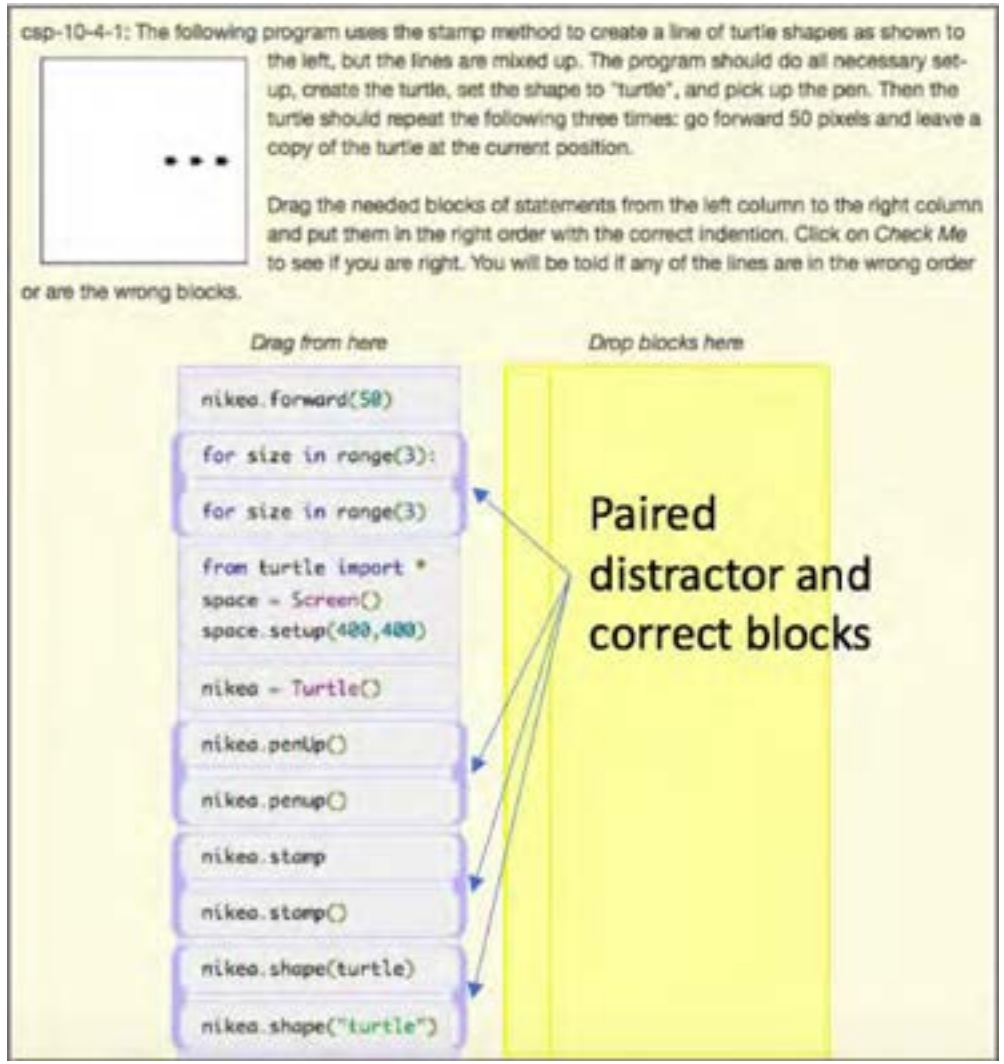
What is a Parsons Problem?

- Provides mixed up code to solve a problem
  - Put the blocks in order
- Can have distractor blocks
  - Common syntax or semantic errors
- Can require indentation

Parsons and Haden 2006

Prior Research on Parsons Problems

- Distractors increase the difficulty
  - Stuart Garner 2007
  - Harms, Chen, and Kelleher 2016
  - Denny, Luxton-Reilly, and Simon 2008
- Pairing the correct and distractor blocks reduces the difficulty
  - Denny, Luxton-Reilly, and Simon 2008
- Providing indentation reduces the difficulty
  - Denny, Luxton-Reilly, and Simon 2008
  - Ihantola and Karavirta 2011
- Less blocks makes the problem easier
  - Denny, Luxton-Reilly, and Simon 2008

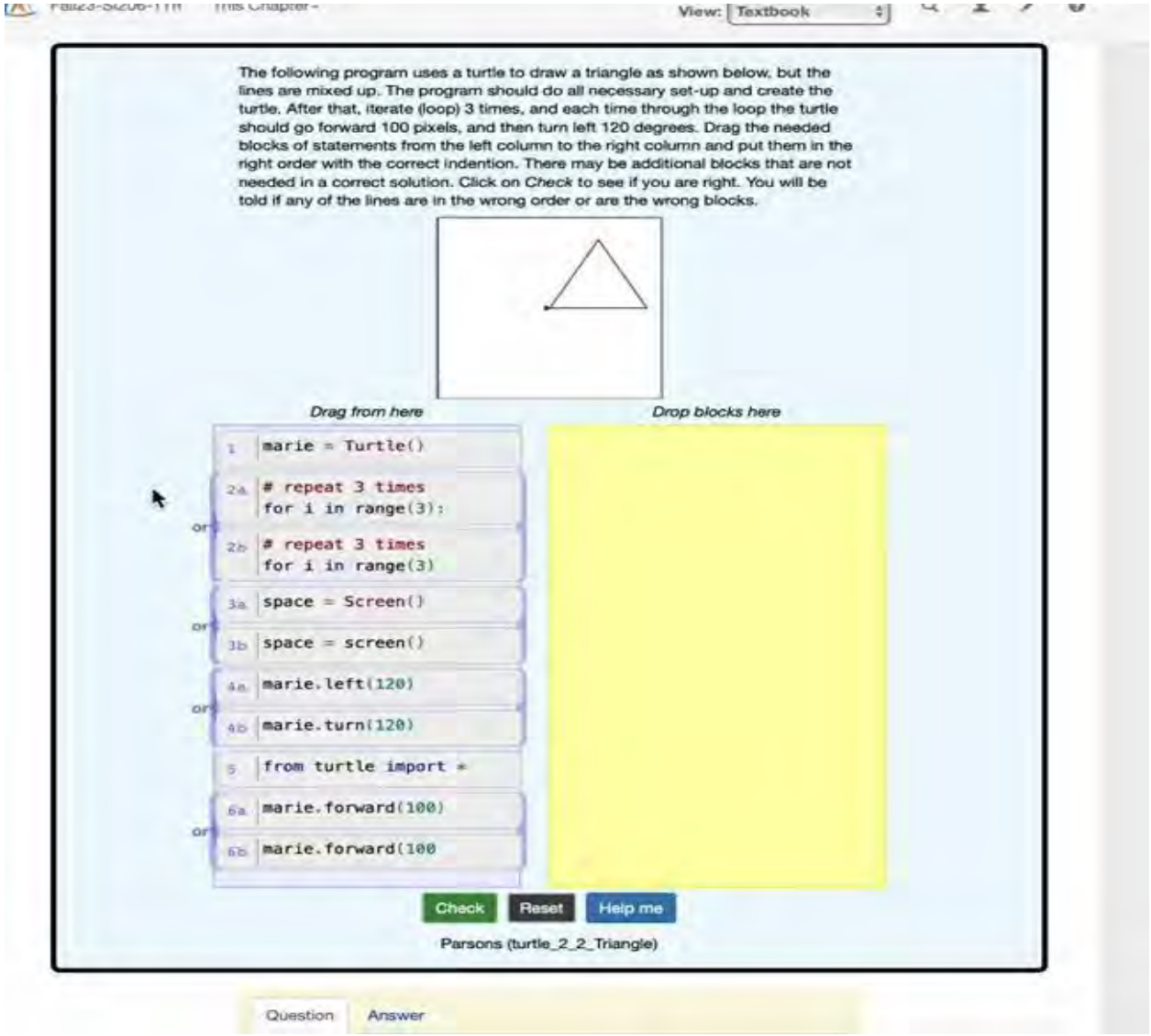


My Team's Research on Parsons Problems

- More people try to solve them than nearby multiple-choice questions
  - *Some people struggle* and *some never solve them*
- Parsons problems are *significantly faster* than fixing and writing code
  - *Equivalent learning gains* from pretest to posttest
- People are nearly *twice as likely to solve* adaptive Parsons
  - Most people prefer the adaptive
- Most students (75+%) find them *useful for learning programming*
  - Some would rather write the equivalent code
- Parsons problems *are not faster* to solve *if the solution is unusual*

Adaptive Parsons Problems

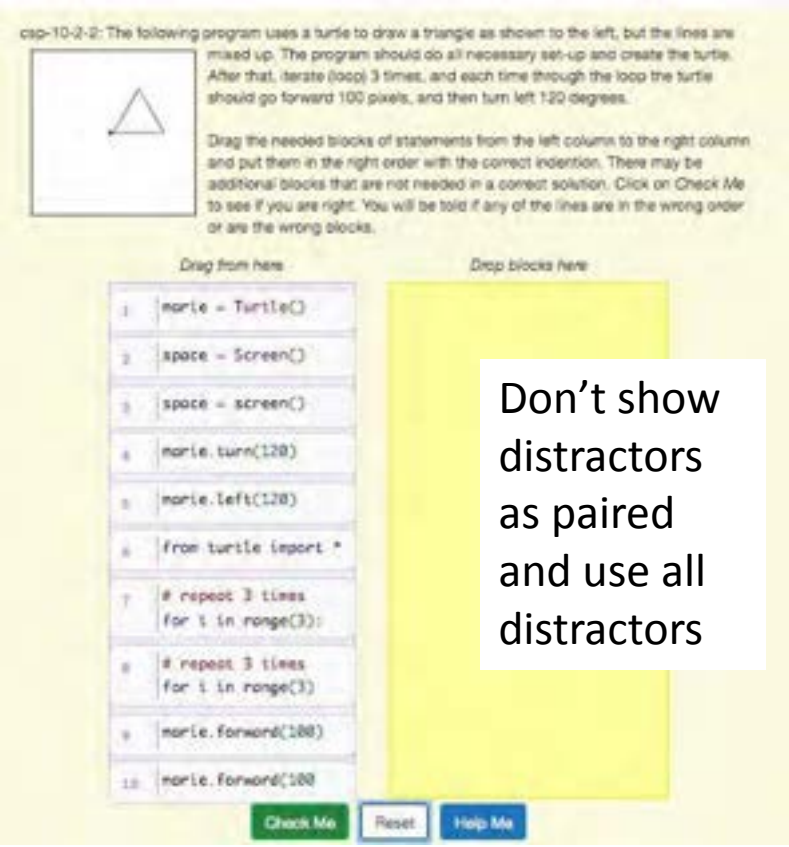
- Intra-problem
  - If the learner is struggling to solve the current problem
    - Remove Distractors
    - Combine Blocks



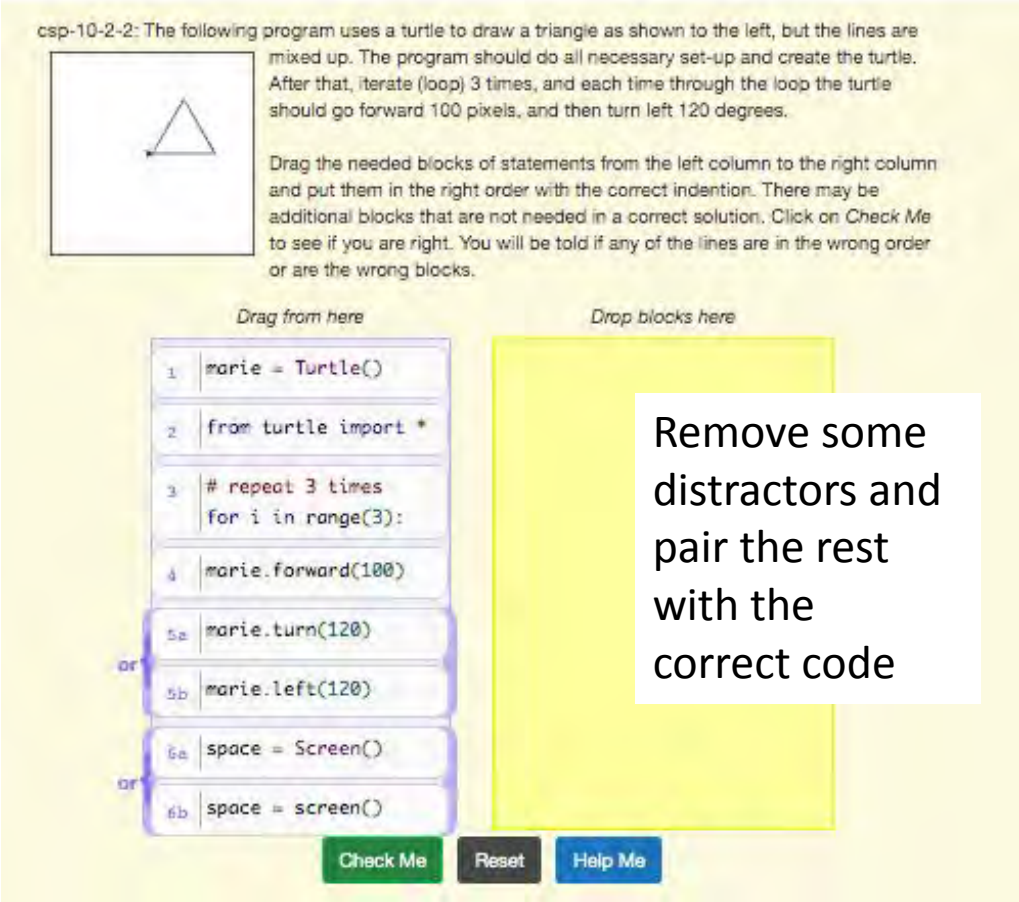
Adaptive Parsons Problems

- Inter-problem
  - If user solved the last one easily, make the next one harder

If many attempts, remove some distractors and pair them with the correct code



Don't show distractors as paired and use all distractors



Remove some distractors and pair the rest with the correct code

Toggle Parsons Problems

- Users can solve either the Parsons problem
  - Or toggle to solve the equivalent write code problem with unit tests
  - Grades whichever one the user leaves selected

Micro Parsons Problems for Regex and SQL

Construct a regex that matches both `cat` and `cats`.

Check Me | Reset

Drag or click the blocks below to form your code:

`s` `cat` `?` `*`

Your code (click on a block to remove it):

`cat` `s` `?`

Activity: 12.11.1 MicroParsons (hp-optional_cats)

Create a student table with an id of type integer (INT), and a name of type string (TEXT).

Check Me | Reset

Drag or click the blocks below to form your code:

`"student"` `"name": TEXT` `(` `"id": INT,` `CREATE` `)`

`"id" INT,` `TABLE` `"name" TEXT`

Your code (click on a block to remove it):

`CREATE` `TABLE` `"student"` `(` `"id" INT,` `"name" TEXT`

Activity: 1 MicroParsons (mparsons-sql-pract-table)

Micro Parsons Problems Research

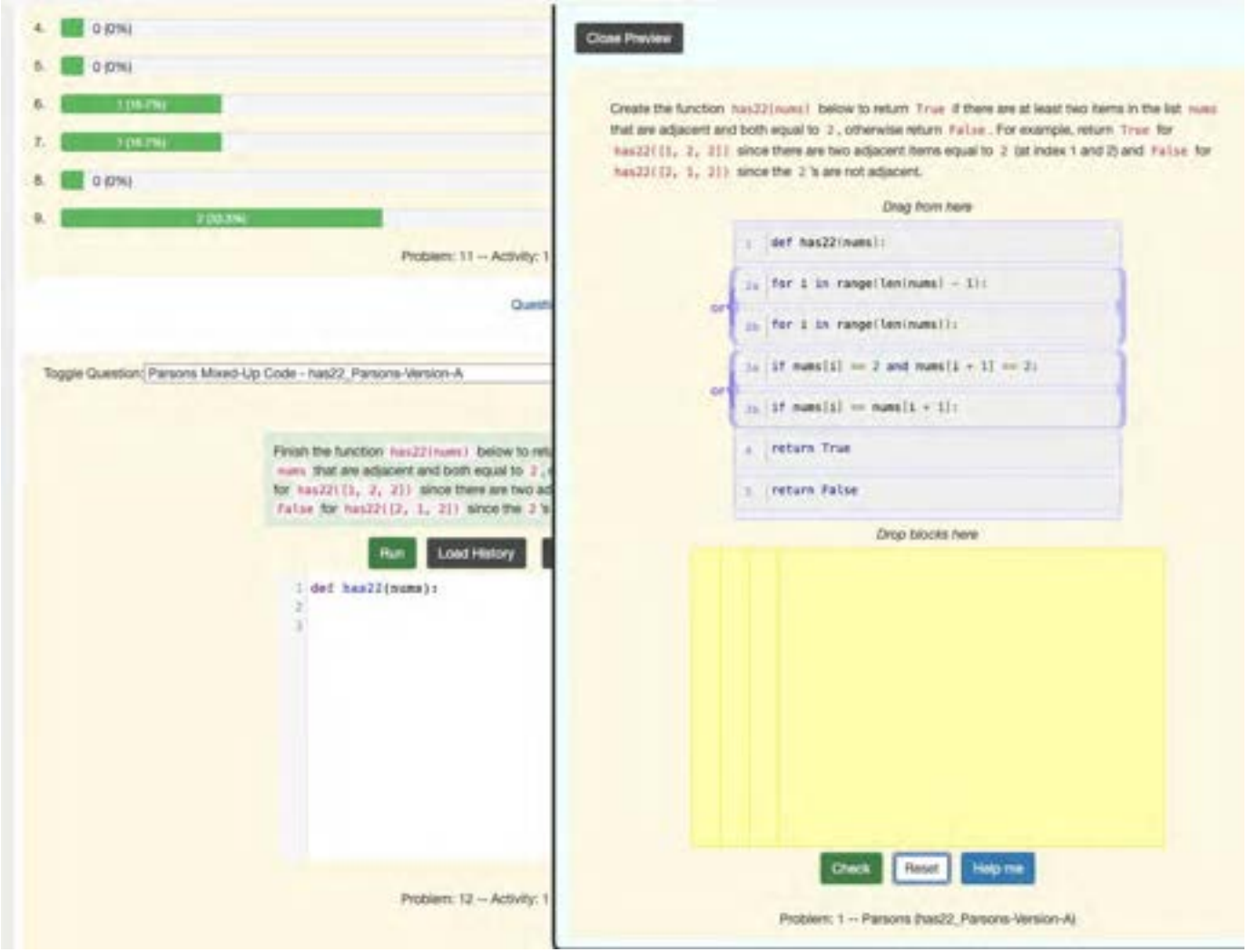
- Micro Parsons condition had a significantly higher completion rate than text-entry
  - and higher learning gains on regex symbols
- Tested two types of feedback with SQL
  - Block based
  - Execution based
- Students who solved SQL micro Parsons in the *blocks-based condition* had *significantly higher learning gain* than in text-entry condition

Wu, Ericson, Brooks 2023

Zihan Wu

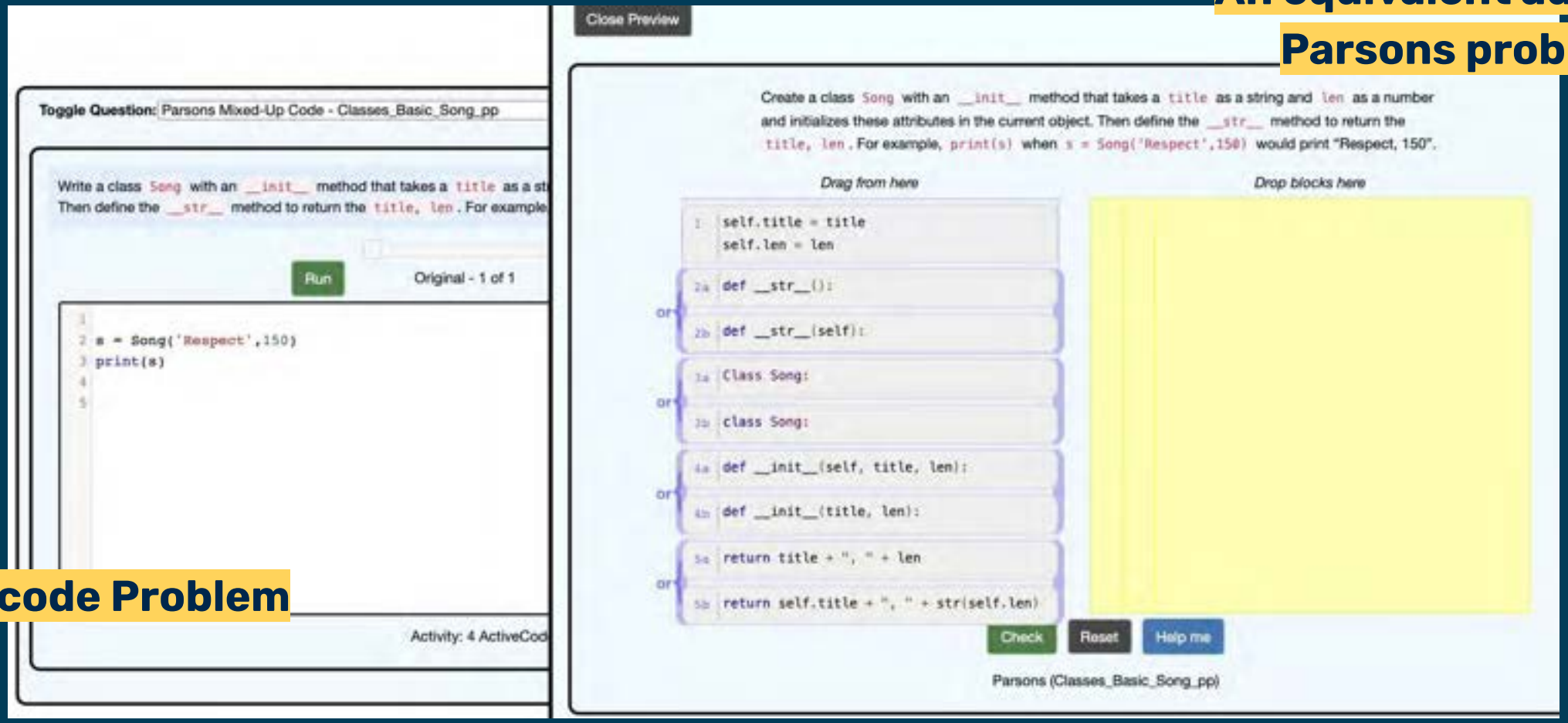
Parsons as Scaffolding While Writing Code

- When writing code
  - Can pop-up a Parsons problem
- Must still solve the write code problem
- Need to analyze log file data to see what percentage of students use this feature



Can Parsons problems support struggling students with completing write-code problems?

An equivalent adaptive Parsons problem



Write-code Problem

Hou, X., Ericson, B. J., & Wang, X. (2022, August). Using adaptive parsons problems to scaffold write-code problems. In *Proceedings of the 2022 ACM Conference on International Computing Education Research-Volume 1* (pp. 15-26).

😄 Most Students Find it Helpful

1 - Reduce difficulty and completion time

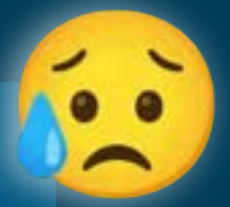
2 - Learn problem-solving strategies

3 - Syntax reminder

4 - Prompt for deeper thinking

Hou et al. 2022 (ICER 2022)

Some Still Struggle to Solve the Parsons Problem



"I was thinking differently on how to approach it, so it's just the Parsons problem (solution) didn't seem reasonable to me." (P9)

"Because a couple of times I used the wrong choice, so that wasn't very helpful. And that kind of led me astray a little bit. " (P3)

Can Parsons support ***align*** with student's approach?

Can Parsons support provide ***targeted*** distractors?

Hou et al. 2022 (ICER 2022)

Students with Low CS Self-efficacy Levels

Practice writing code with Parsons problem or independently

🔼 Higher Practice Performance

🔼 Higher Problem-solving Efficiency

When practice writing code with Parsons problem

More likely to ***solve*** the Parsons problem

Can Parsons support be more ***concise*** in a personalized way?

Hou, X., Ericson, B. J., & Wang, X. (2023). Understanding the Effects of Using Parsons Problems to Scaffold Code Writing for Students with Varying CS Self-Efficacy Levels. Proceedings of the 23rd Koli Calling International Conference on Computing Education Research

Hou et al. 2023 (Koli Calling 2023)

2022-2023:

GitHub Copilot, Open AI ChatGPT,

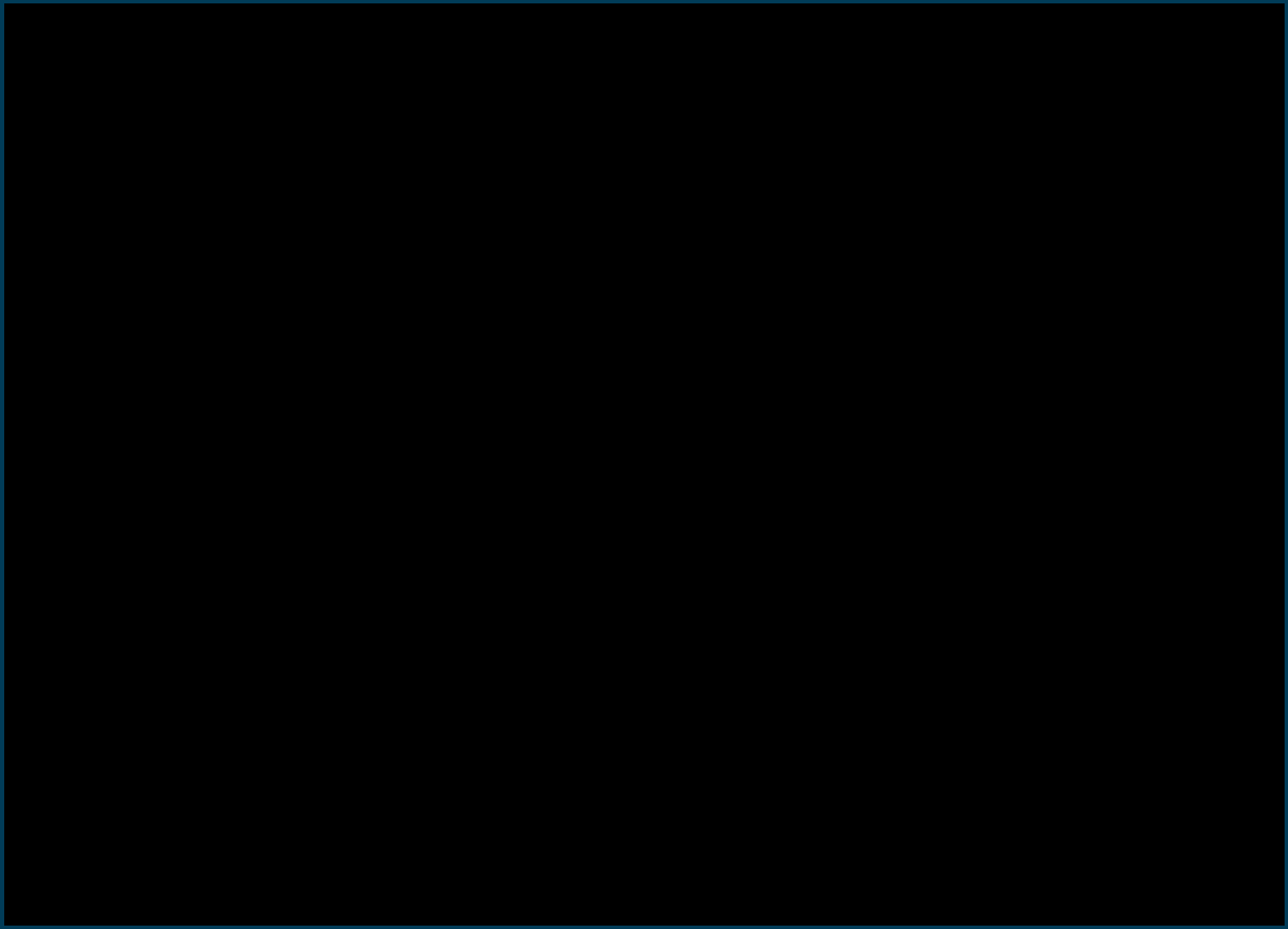
Google Bard, Anthropic Claude ...

Generative AI tools can generate <u>code solutions</u>

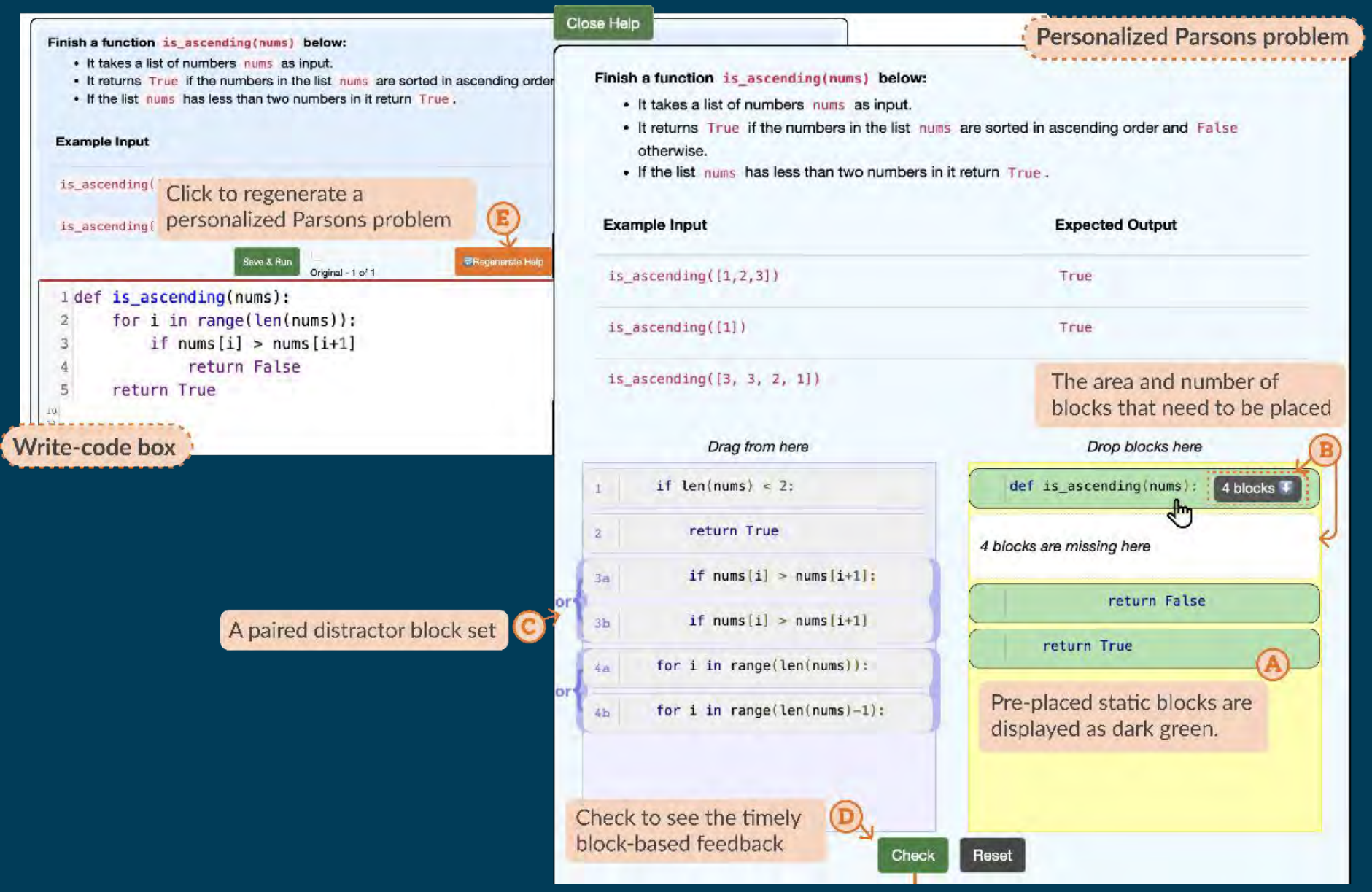
They are <u>now in CS classrooms</u>

I don't know how to solve the problem.

So I'll just use ChatGPT and submit the work!



CodeTailor



Hou et al. 2024 (Arxiv soon)

Scenario 1: Receive a fully movable Parsons problem with "combine block" feature and no distractors

```
1 def is_ascending(nums):
2     if len
3
4
5
6
7
8
9
10
```

P1 Generate movable Parsons blocks based on a correct solution

Drag from here

```
if (len(nums) < 2):
for i in range(len(nums) - 1):
return True
def is_ascending(nums):
    if (nums[i] > nums[i+1]):
        return False
    return True
```

Drop blocks here

click to combine two blocks into one block

k

Check | Reset | Combine Blocks

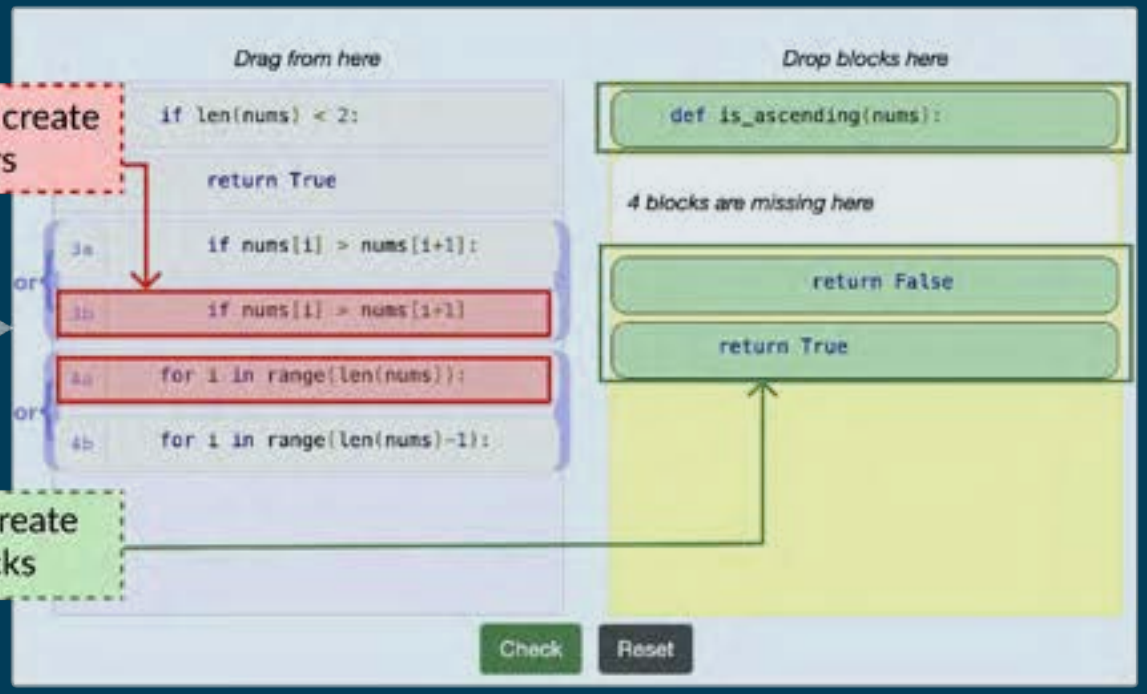
Hou et al. 2024 (Arxiv soon)

Scenario 2: Receive a partially movable Parsons problem with paired distractors from student errors

```
1  def is_ascending(nums):
2      for i in range(len(nums)):
3          if nums[i] > nums[i+1]
4              return False
5      return True
6
7
8
9
10
```

P2 Use incorrect lines to create paired distractors

P3 Use correct lines to create static Parsons blocks



Hou et al. 2024 (Arxiv soon)

Scenario 3: Receive a partially movable Parsons problem with paired distractors from AI and student errors

```
1  def is_ascending(nums):
2      if len(nums) < 2:
3          return True
4      i = 0
5      while i < len(nums):
6          if nums[i] > nums[i+1]:
7              return False
8          i += 1
9      return True
10
```

Use the incorrect line to create a paired distractor

P4 Convert a correct line into a distractor generated by LLM.

Drag from here

1a `i = 0`
`while i < len(nums) - 1:`

or

1b `i = 0`
`while i < len(nums):`

2a `if (nums[i] < nums[i+1]):`

or

2b `if nums[i] > nums[i+1]:`

6 static Parsons blocks from correct lines

Drop blocks here

```
def is_ascending(nums):
if len(nums) < 2:
return True
```

2 blocks are missing here

```
return False
i += 1
return True
```

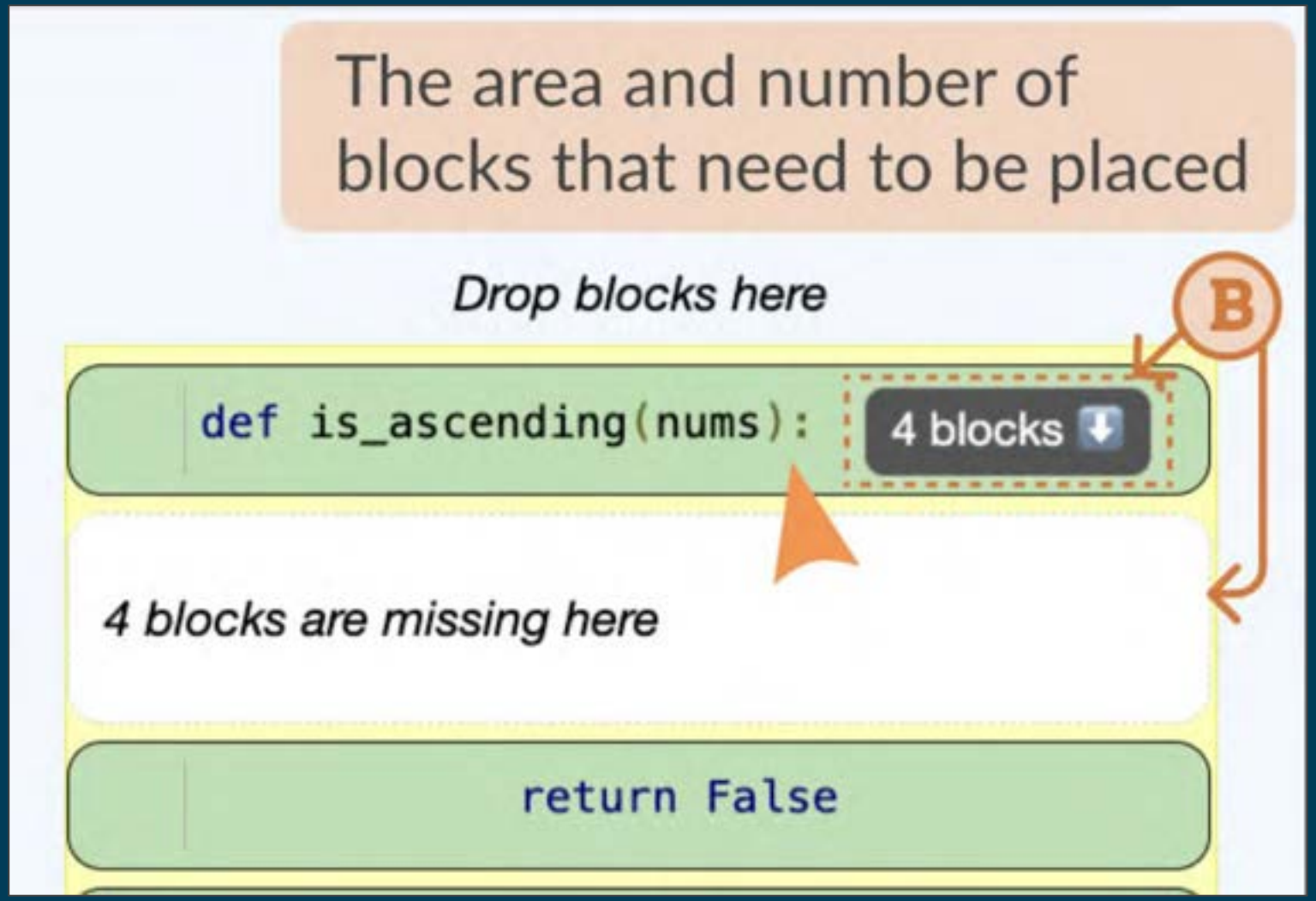
Check Reset

Hou et al. 2024 (Arxiv soon)

Block Relative Placement Guidance

Timely Block-based Feedback

Regenerate the Help



Hou et al. 2024 (Arxiv soon)

Block Relative Placement Guidance

Timely Block-based Feedback

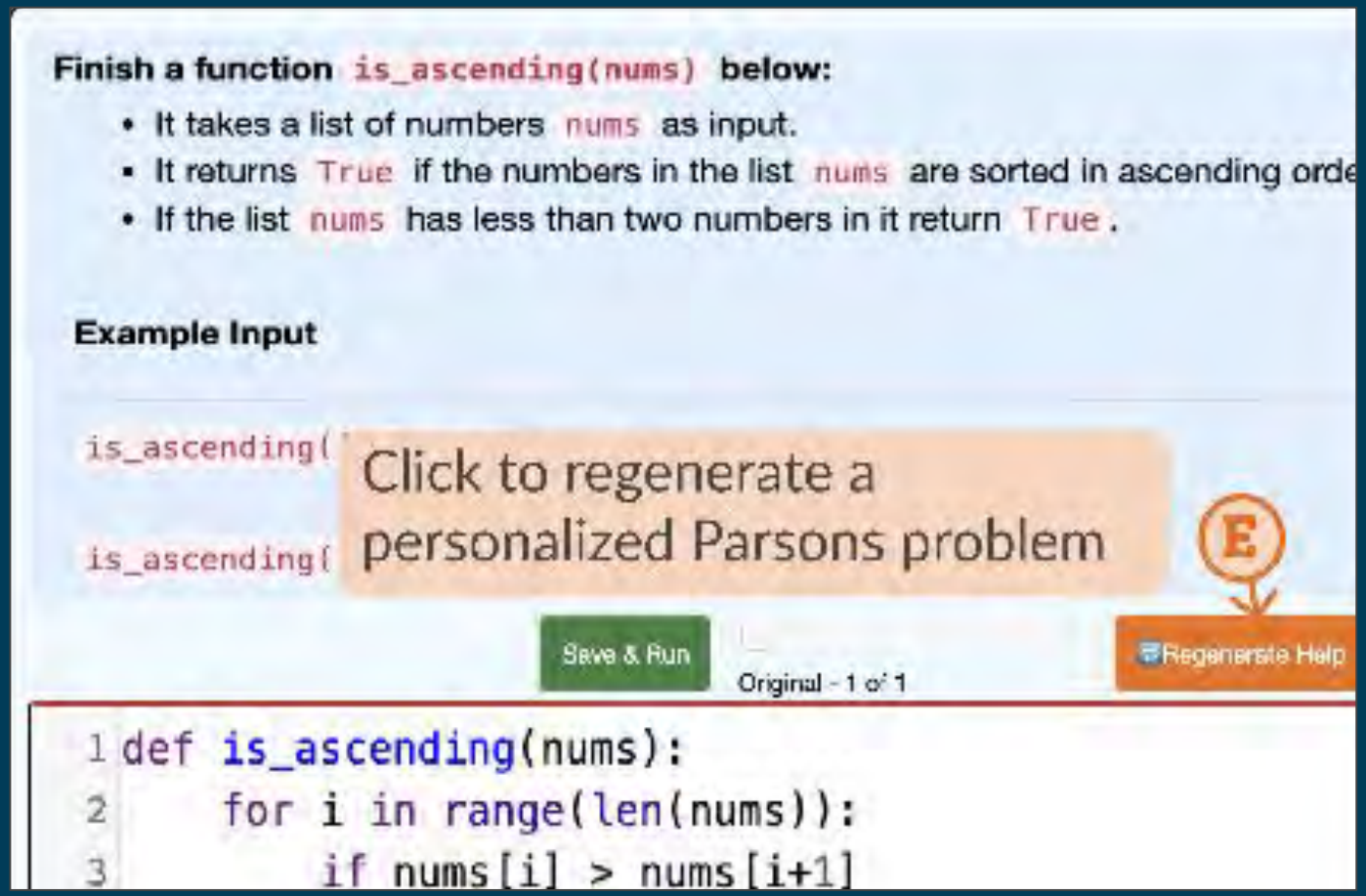
Regenerate the Help

Hou et al. 2024 (Arxiv soon)

Block Relative Placement Guidance

Timely Block-based Feedback

Regenerate the Help



Hou et al. 2024 (Arxiv soon)

In 2023 summer, with …

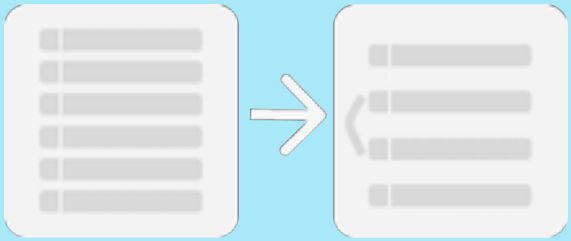
- **800 former student buggy code:** Can CodeTailor provide high-quality personalized Parsons puzzles?
- **18 novice programming learners:** Can CodeTailor support a more <u>engaging</u> and <u>educational</u> write-code experience than the typical AI-generated solution?

Hou et al. 2024 (Arxiv soon)

😄 CodeTailor can

Error-Free Solution Generation

Closely align with the incorrect student code compared to a common solution.



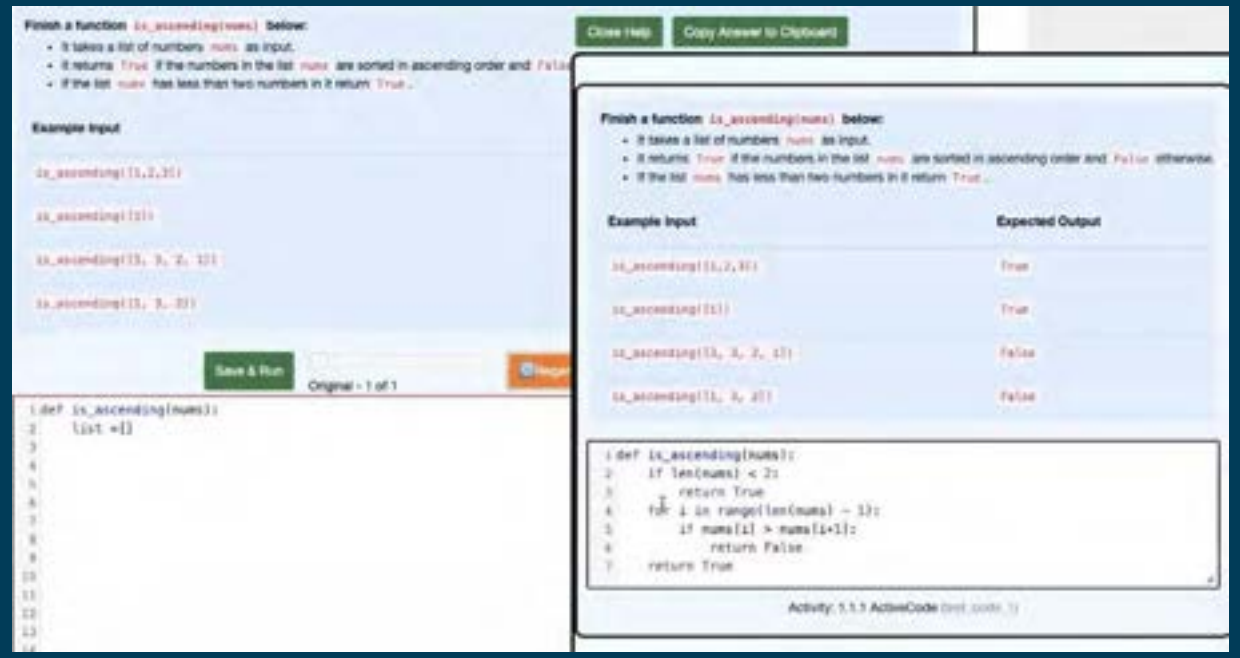
Delivery of More Concise Parsons Problems

Hou et al. 2024 (Arxiv soon)

😄 Students …
(CodeTailor / An AI-generated solution)

Report higher <u>engagement</u> with CodeTailor

<u>Recalled and applied more</u> new components from the practice to the posttest



Hou et al. 2024 (Arxiv soon)

😄 Students …
(CodeTailor / An AI-generated solution)

More interactive

Right amount of effort

Think of the solution-building process

88% preferred CodeTailor more for <u>learning</u>

Boost confidence

Diagnose original thinking

Hou et al. 2024 (Arxiv soon)

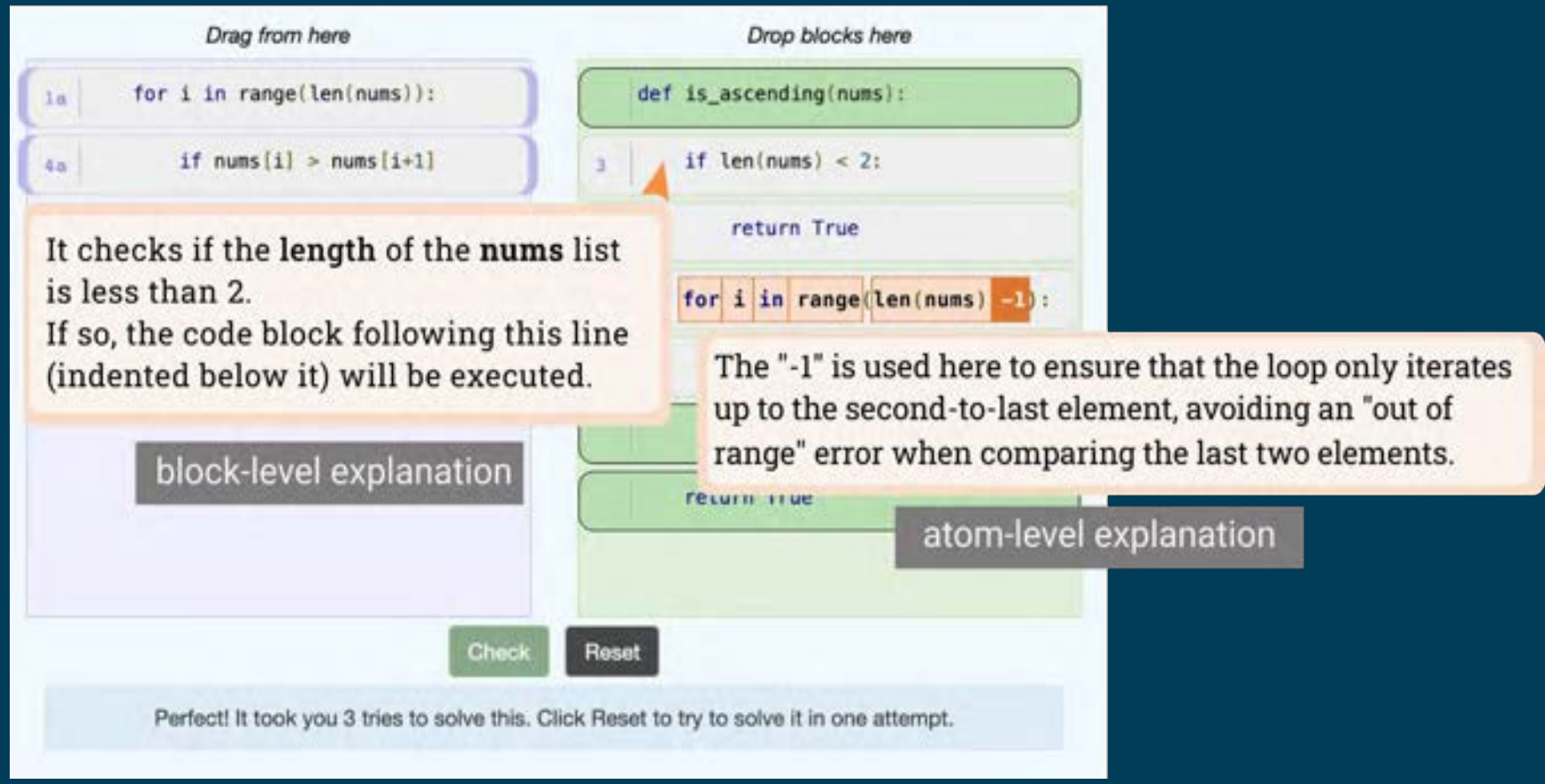
😥 Students with CodeTailor

Can <u>solve</u>, but not <u>fully understand</u> the solved solution

Hou et al. 2024 (Arxiv soon)

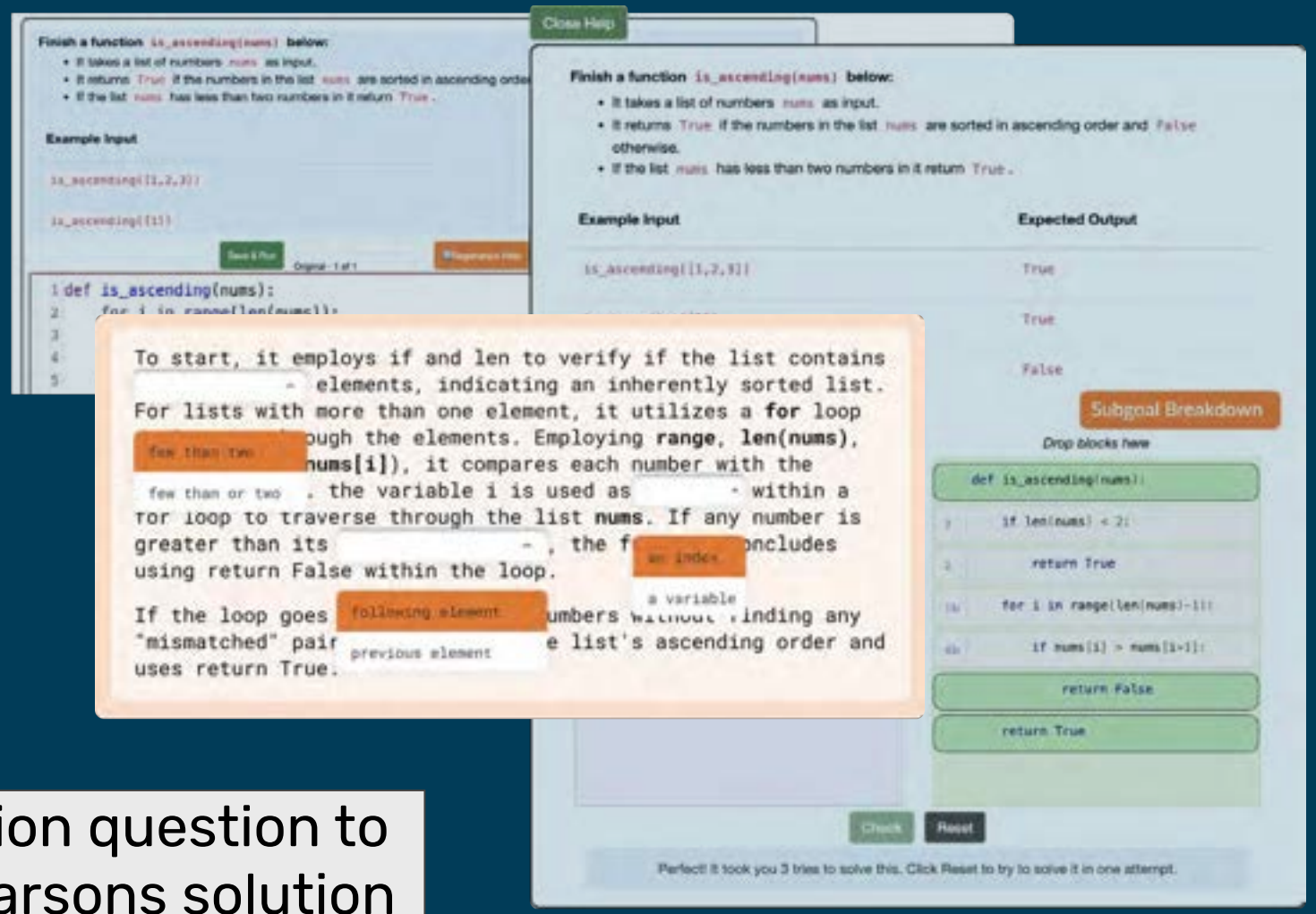
To address this challenge …

Subgoal guidance explanation



Hou et al. 2024 (SIGCSE 2024)

In 2023 fall, to address this challenge …

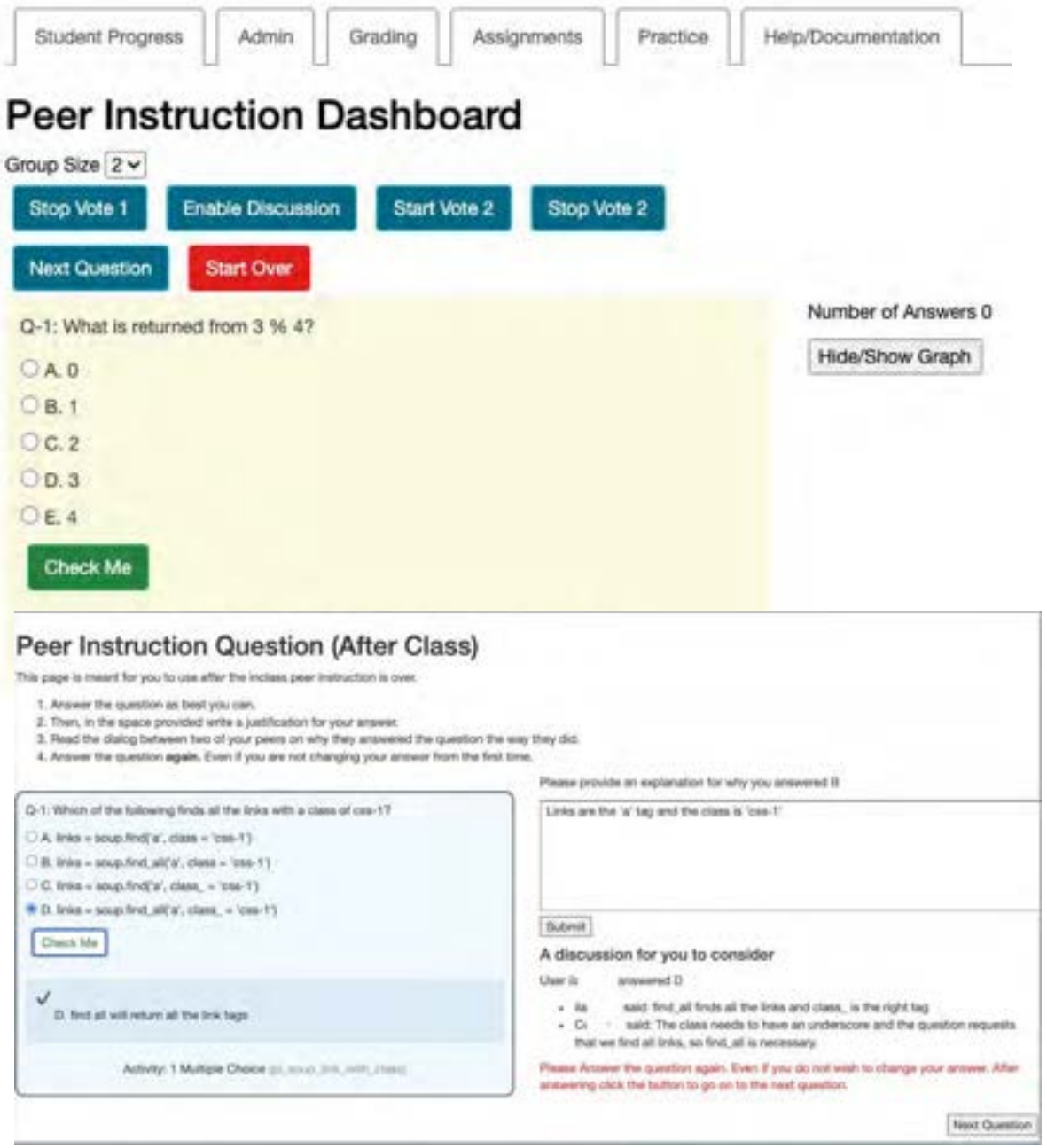


A self-explanation question to reflect on the Parsons solution

Hou et al. 2024 (SIGCSE 2024)

Peer Instruction

- Created by Eric Mazur
  - Display a hard multiple-choice question
  - Students vote individually
  - Discuss with peers
  - Vote again
- More students succeed
  - But clickers are problamatic
  - Hard to find good questions
- We have added existing questions to ebooks
- Added new interfaces
  - In lecture text-chat
  - After lecture pseudo text-chat



Guided Inquiry Learning in Lecture

- Work in groups
  - One person fills in answers and shares with others
- Activities expose students to the concepts to be learned
- Students record what they learned and any questions in padlet
- Barb goes over questions from the padlet

https://tinyurl.com/bddz8zx3

5.29. Group Work: Functions and Lists

It is best to use a POGIL approach with the following. In POGIL students work in groups on activities and each member has an assigned role. For more information see https://cspogil.org/Home.

Note

If you work in a group, have only one member of the group fill in the answers on this page. You will be able to share your answers with the group at the bottom of the page.

Learning Objectives

Students will know and be able to do the following.

Content Objectives:

- Use positive and negative indices to access elements of a list.
- Identify the purpose of common list methods and common methods that take lists as parameters
- Use the slice operator to copy parts of a list.

Process Objectives:

- Predict the output of code with lists (Information Processing)
- Write code using the slice operator (Assessment)

5.29.1. List Indexing

A list holds items in order and you can get the value at an index, just like you can with strings.

Q-1: What is the last thing that will be printed when the code below runs?

hi

Check me | Compare me

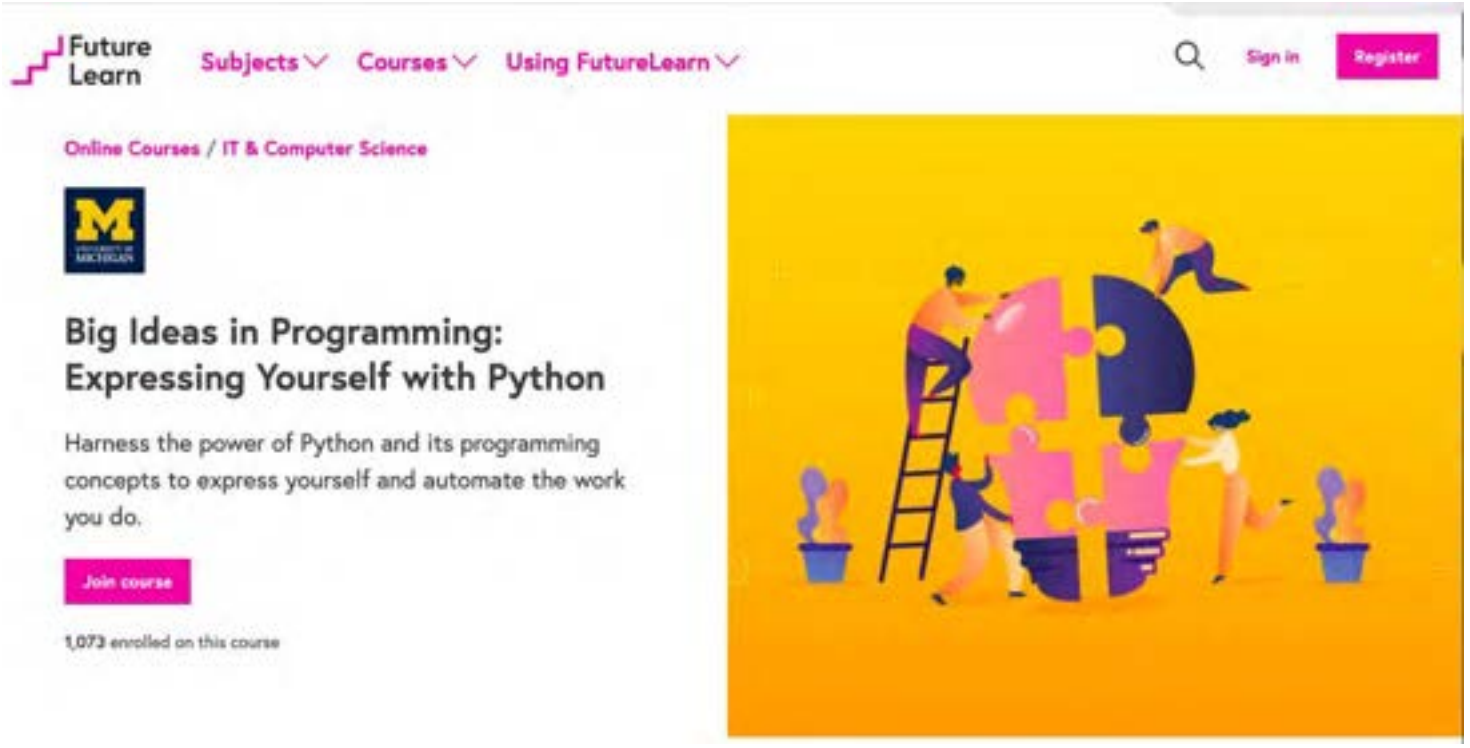
Social Justice in Computing

- Semi-structured interviews with 11 students
  - 63% say computing lacks communal applications
- Most students wanted more homework/projects with a social justice context
  - Real data, direct action, in-class discussion
- Use satirical Programming to critique computing artifacts

Aadarsh Padiyath

MOOC Based on Python for Everybody

- Four courses will be on Courserea in late Feb
- Audience
  - teachers
  - students
  - conversational programmers
- Not lecture!
  - Open-ended and creative assignments
  - Work in interactive ebook



Summary

http://tinyurl.com/ericsonJan2024

- Use active learning
  - Free and interactive ebooks
  - Peer Instruction
  - Low cognitive load practice (Parsons problems)
  - Process oriented guided inquiry learning (POGIL)
- Use AI with guardrails to help students succeed
- Use AI to generate or personalize problems
- Incorporate social justice in computing

Runestone Ebooks