Using LLMs to Generate Personalized Assignments and Explanations

Dr. Barbara Ericson

Assistant Professor at the School of Information and the College of Engineering

barbarer@umich.edu

The Need for Student Support

Many students struggle when developing cognitive skills

- Math
- Science
- Programming

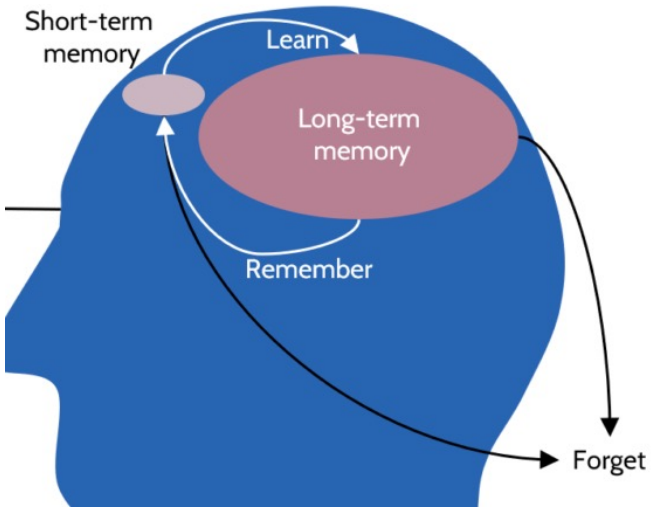
Failure, especially early when learning a new skill, reduces self-efficacy

Practice is Crucial for Learning

- Must be successful and include timely feedback
- Interactive > Constructive > Active > Passive

  Chi and Wylie, *The ICAP framework: Linking cognitive engagement to active learning outcomes*, 2014
- Spaced practice is better for long term learning
- Desirable difficulties can reduce learning in the short-term
  - But increase long-term learning

Cognitive Load Theory



Humans process new information in working memory

- Limited capacity
- Whole tasks can impede learning
- Completion tasks can reduce cognitive load
- Worked examples plus practice best
- Avoid expertise reversal effect

Sweller, *Cognitive load during problem solving: Effects on learning*, 1988
Sweller, van Merriënboer, Paas, *Cognitive architecture and instructional design: 20 years later*, 2019

Parsons Problems

Create a class `Song` with an `__init__` method that takes a `title` as a string and `len` as a number and initializes these attributes in the current object. Then define the `__str__` method to return the `title, len`. For example, `print(s)` when `s = Song('Respect',150)` would print "Respect, 150".

Drag from here

```
1a  def __init__(title, len):
or
1b  def __init__(self, title, len):

2a  Class Song:
or
2b  class Song:

3   self.title = title
    self.len = len

4a  def __str__(self):
or
4b  def str(self):

5a  return self.title + ", " + str(self.len)
or
5b  return title + ", " + len
```

Drop blocks here

Solution

```
2b  class Song:
    1b  def __init__(self, title, len):
        3   self.title = title
            self.len = len
        4a  def __str__(self):
            5a  return self.title + ", " + str(self.len)
```

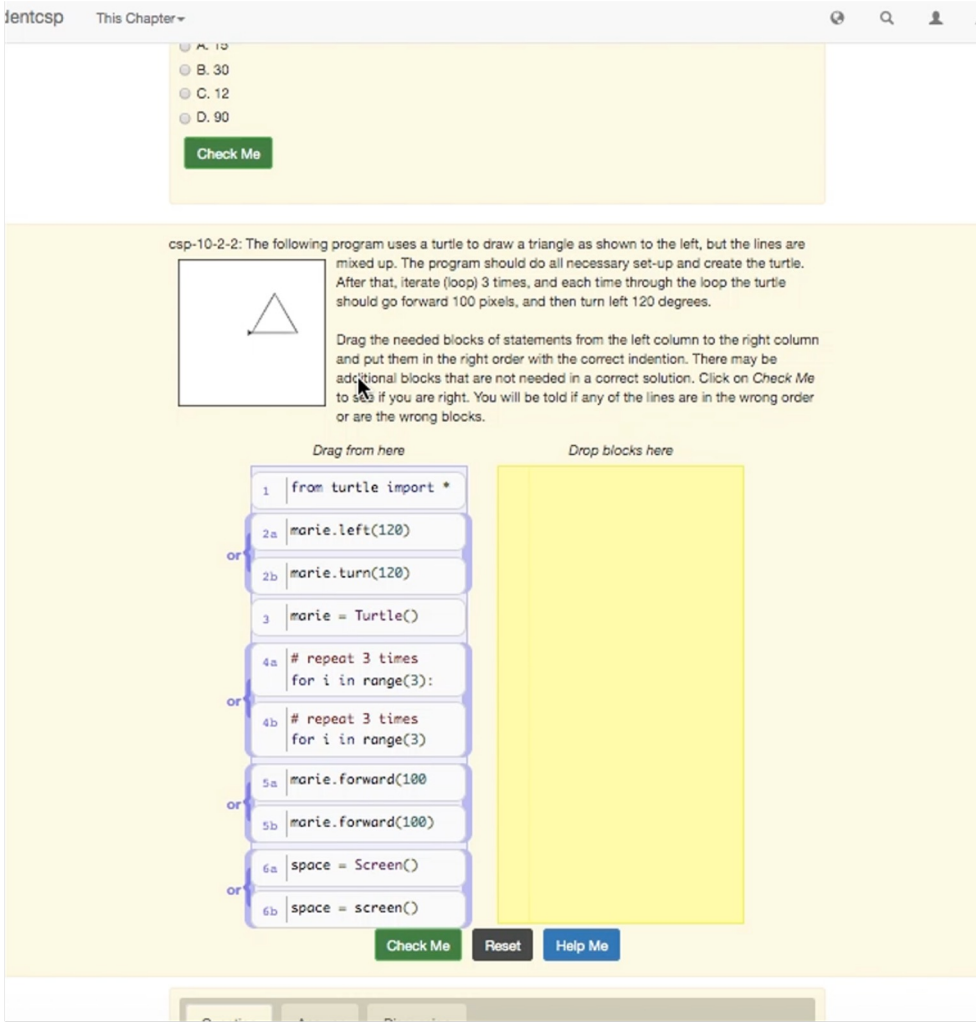
Check | Reset | Help me

Parsons (Classes_Basic_Song_wd3_pp)

- Completion problems
- Usually have lower cognitive load than writing code
- Many variants
  - Adaptive
  - Proof Blocks
  - Faded
  - Micro

Adaptive Parsons Problems

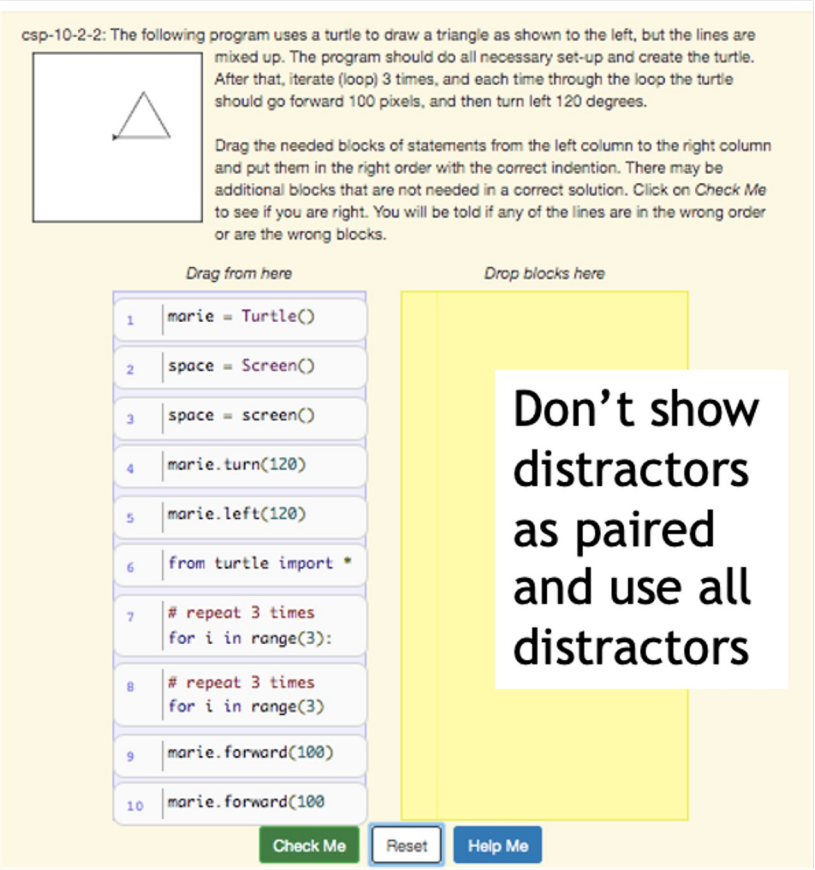
- Intra-problem
  - If the learner is struggling to solve the current problem
    - Remove distractors
    - Provide Indentation
    - Combine Blocks



Adaptive Parsons Problems

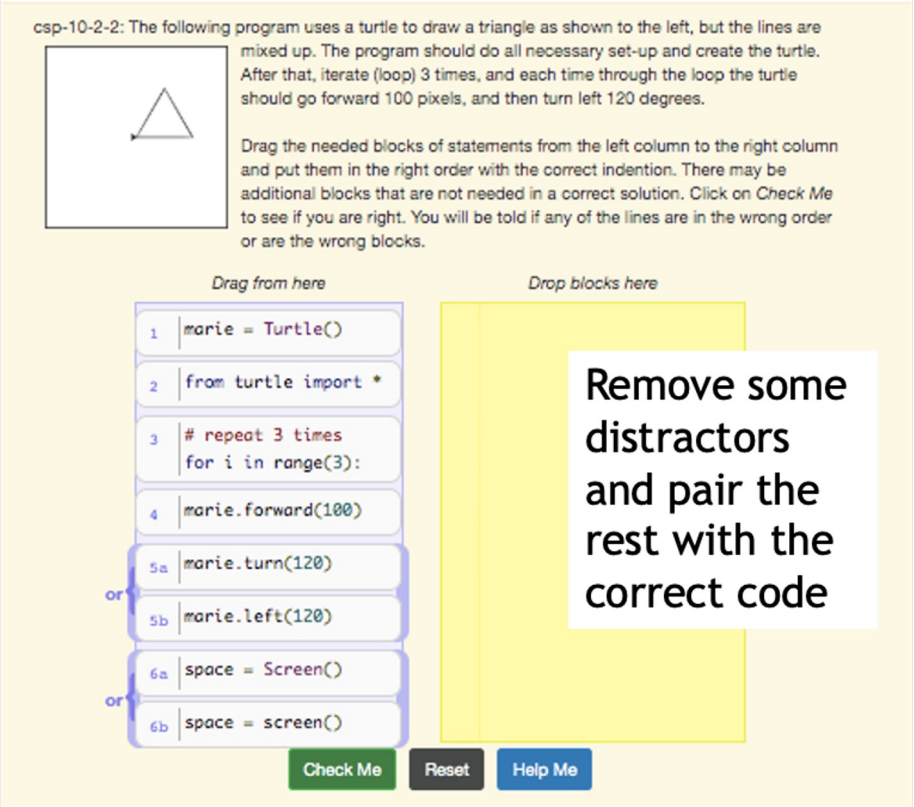
- Inter-problem

  If solved the last one easily make the next one harder



Don't show distractors as paired and use all distractors

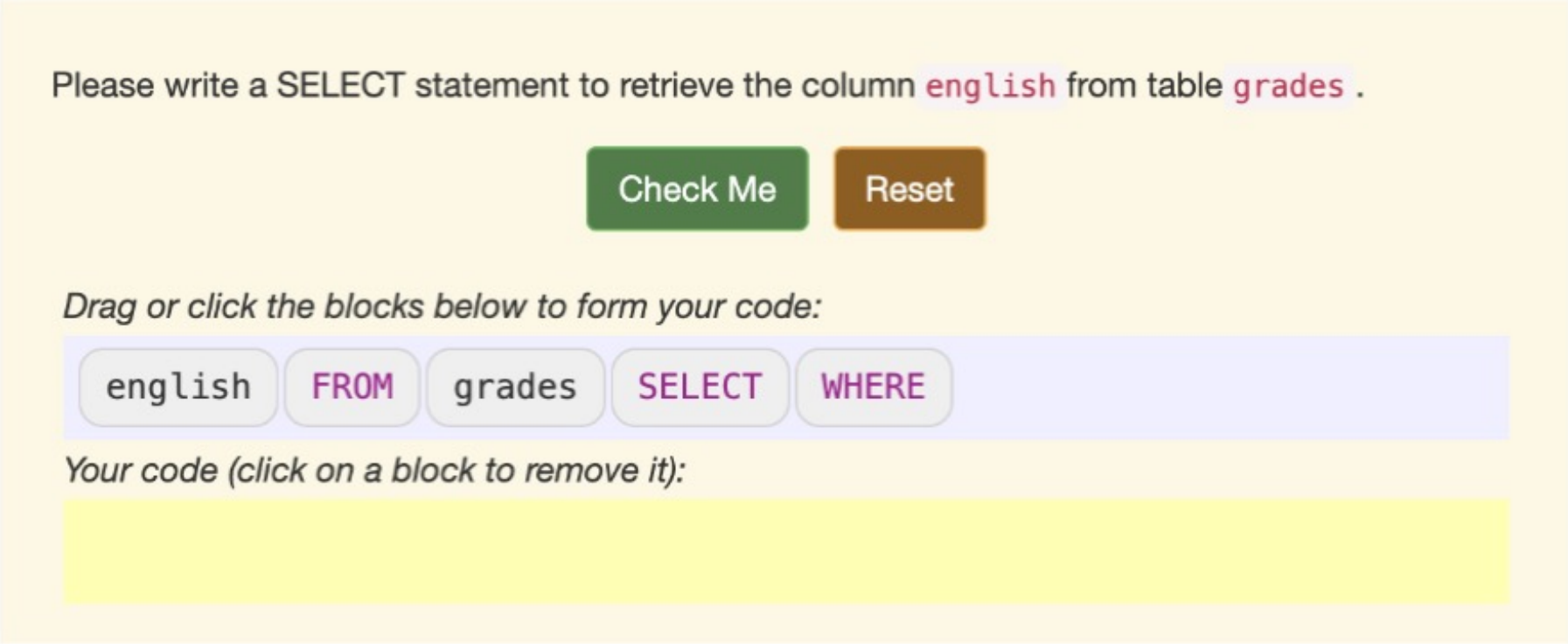
If many attempts then remove some distractors and pair them with the correct code



Remove some distractors and pair the rest with the correct code

Micro Parsons – Zihan Wu

- Assemble fragments into order to create a statement



Research on Parsons Problems

- Most learners find them useful
- Significantly faster to solve than writing the equivalent code
  - With similar learning gains from pre to post
- Significantly more learners complete them vs write code problems in voluntary practice
- Can be used to learn common algorithms
- Nearly twice as likely to correctly solve adaptive than non-adaptive

Ericson et at, Parsons Problems and Beyond: Systematic Literature Review and Empirical Study Designs, 2022

Issues with Parsons Problems

Some learners would rather write the code themselves

- Especially those with more prior experience
- View writing code as more authentic

Some can solve them (especially adaptive)

- But not understand the solution

Giving Students A Choice

If given a Parsons problem

- Can switch to the equivalent write code
- Grade whichever is left on the page

Toggle Question: Parsons Mixed-Up Code - mult_class_point_and_triangle_practice_pp

Given a `Point` class that has a `distance` object method that takes another point and returns the distance between the two points, create a `Triangle` class with an `__init__` method that takes three points. Also write the `perimeter` method in the `Triangle` class to return the sum of the lengths of the sides of the triangle. Use the `distance` method in the `Point` class to calculate the distance between two points.

Drag from here | *Drop blocks here*

```
1   class Triangle:
2   return total
3a  def __init__(p1, p2, p3):
or
3b  def __init__(self, p1, p2, p3):
4a  total = distance(self.p1, self.p2)
or
4b  total = self.p1.distance(self.p2)
5   self.p1 = p1
    self.p2 = p2
    self.p3 = p3
6a  total += self.p2.distance(self.p3)
    total += self.p3.distance(self.p1)
or
6b  total += distance(self.p2,
    self.p3)
    total += distance(self.p3,
    self.p1)
7a  def perimeter():
or
7b  def perimeter(self):
```

Check | Reset | Help me

Parsons (mult_class_point_and_triangle_practice_pp)

Toggle Question: Active Write Code - mult_class_point_and_triangle_practice_ac

Given a `Point` class below with a `distance` object method that takes another point and returns the distance between the two points, create a `Triangle` class with an `__init__` method that takes three points. Also write the `perimeter` method in the `Triangle` class to return the sum of the lengths of the sides of the triangle. Use the `distance` method in the `Point` class to calculate the distance between two points.

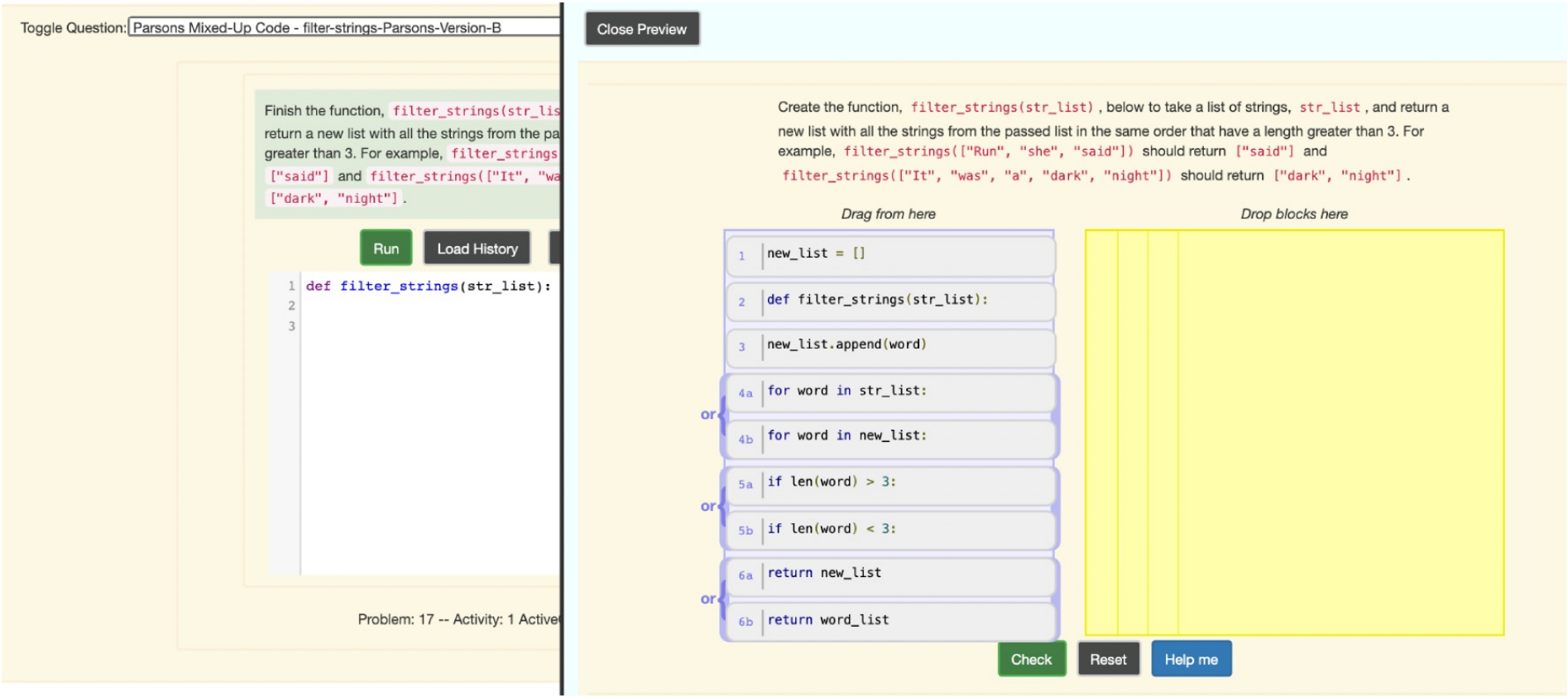
Run | Original - 1 of 1 | Show CodeLens | Share Code

```
1  import math
2  class Point:
3
4      def __init__(self, initX, initY):
5          """ Create a new point at the given coordinates. """
6          self.x = initX
7          self.y = initY
8
9      def distanceFromOrigin(self):
10         return ((self.x ** 2) + (self.y ** 2)) ** 0.5
11
12     def __str__(self):
13         return f"x = {self.x}, y = {self.y}"
14
15     def halfway(self, target):
16         mx = (self.x + target.x) / 2
17         my = (self.y + target.y) / 2
18         return Point(mx, my)
19
20     def distance(self, other):
21         mx = (self.x - other.x) ** 2
22         my = (self.y - other.y) ** 2
23         return math.sqrt(mx + my)
24
25 class Triangle:
26
27
```

Activity: 5 ActiveCode (mult_class_point_and_triangle_practice_ac)

Parsons as Scaffolding

Students can pop-up an equivalent Parsons problem while writing code



Parsons as Scaffolding Results

A think-aloud study with 11 students

- Use to get started
- Help when stuck (how do I do *x* in Python)
- Use to debug

Between-subject study with 81 undergraduate students

- Parsons group took significantly less time to complete practice problems
- However, there was a ceiling effect on the pre-test

Hou, Ericson, and Wang, *Using Adaptive Parsons Problems to Scaffold Write-Code Problems*, ICER, 2022

Parsons as Scaffolding Results

Between-subjects study with 89 undergraduates

- Parsons as scaffolding vs No scaffolding (write code)
- Students with low computing self-efficacy
  - With scaffolding had significantly higher practice performance and practice efficiency than no scaffolding
  - Still had students who didn’t understand the Parsons solution
  - Parsons not helpful if it didn’t match their approach

Hou, Ericson, and Wang, Koli, 2023

ActiveHint – Xinying Hou

Use an LLM to generate a Parsons problem from a student's incorrect solution

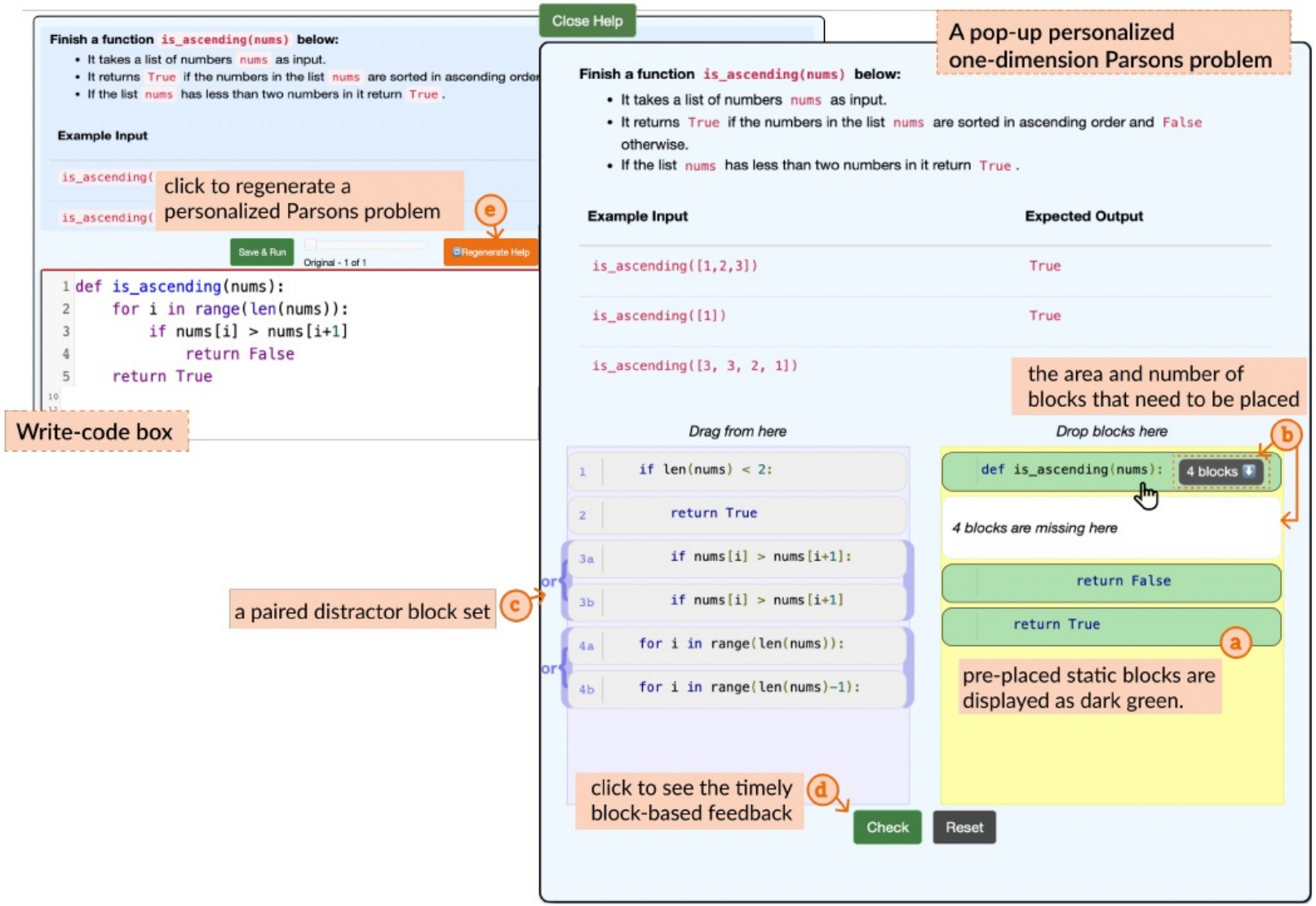
1) Technical evaluation

   Does ActiveHint create a more similar Parsons problem than using the most common solution?

2) Within-subjects study with 18 undergraduates. Received both generated solution and a personalized Parsons problem.

Not yet published

Personalized Parsons Problems



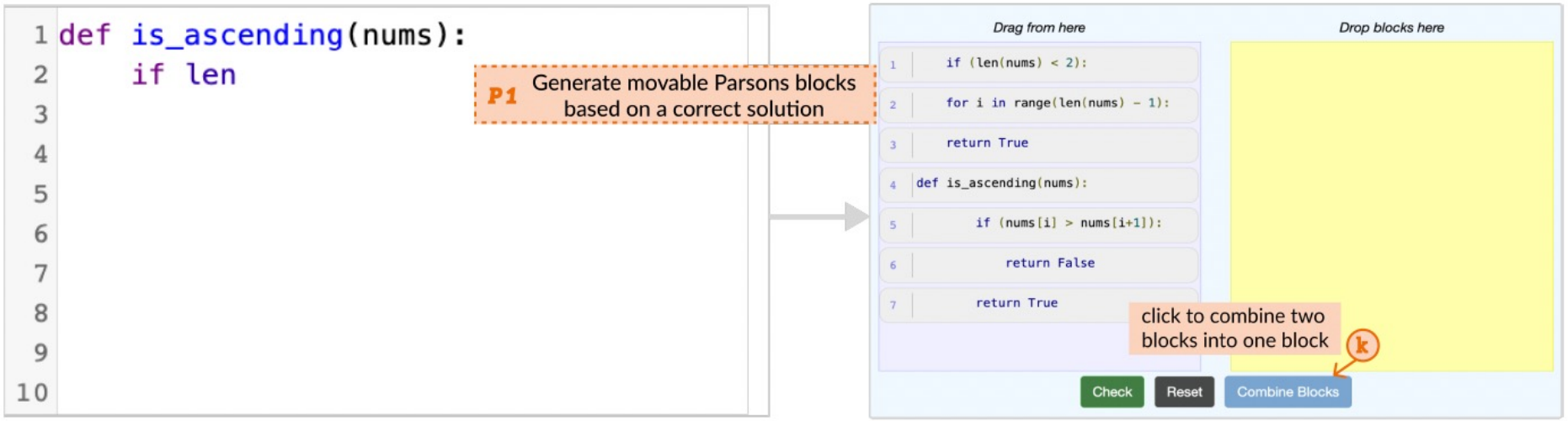
Use student's incorrect solution and an LLM to generate a personalized Parsons problem

Use student's incorrect code as distractors

User Has Not Written Anything/Much

- Generate Parsons from most common solution

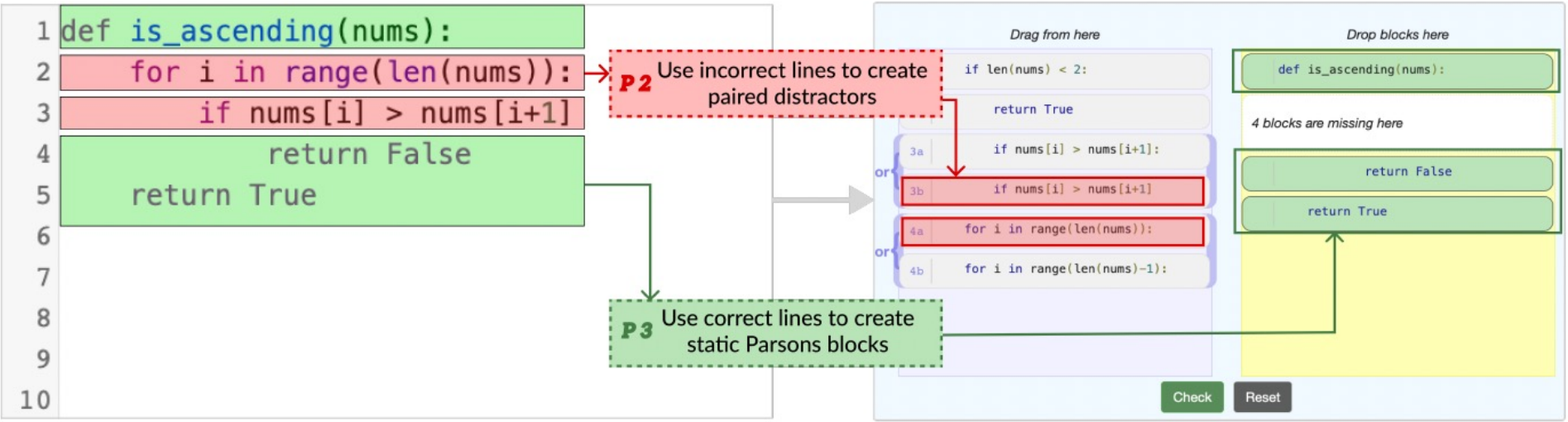
***A**: Receive a fully movable Parsons problem that includes the combine block feature and no distractors*



User Has Written Incorrect Code

- Generate distractors from the incorrect parts

***B**: Receive a partially movable Parsons problem with paired distractors containing student incorrect lines.*



Getting Help

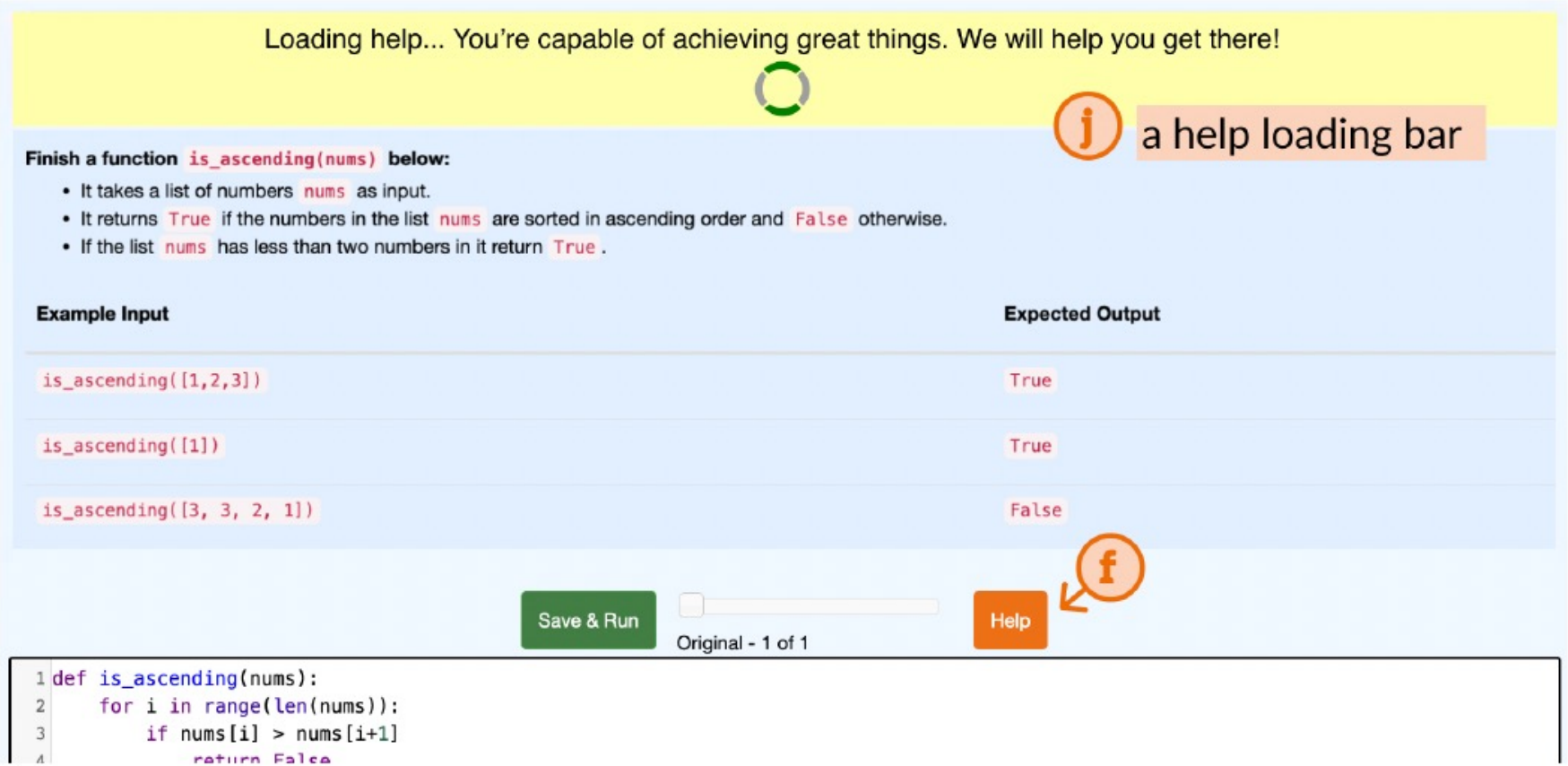


Fig. 2. The help loading bar of ActiveHint includes a spinning loader and an encouragement sentence

Generation Process

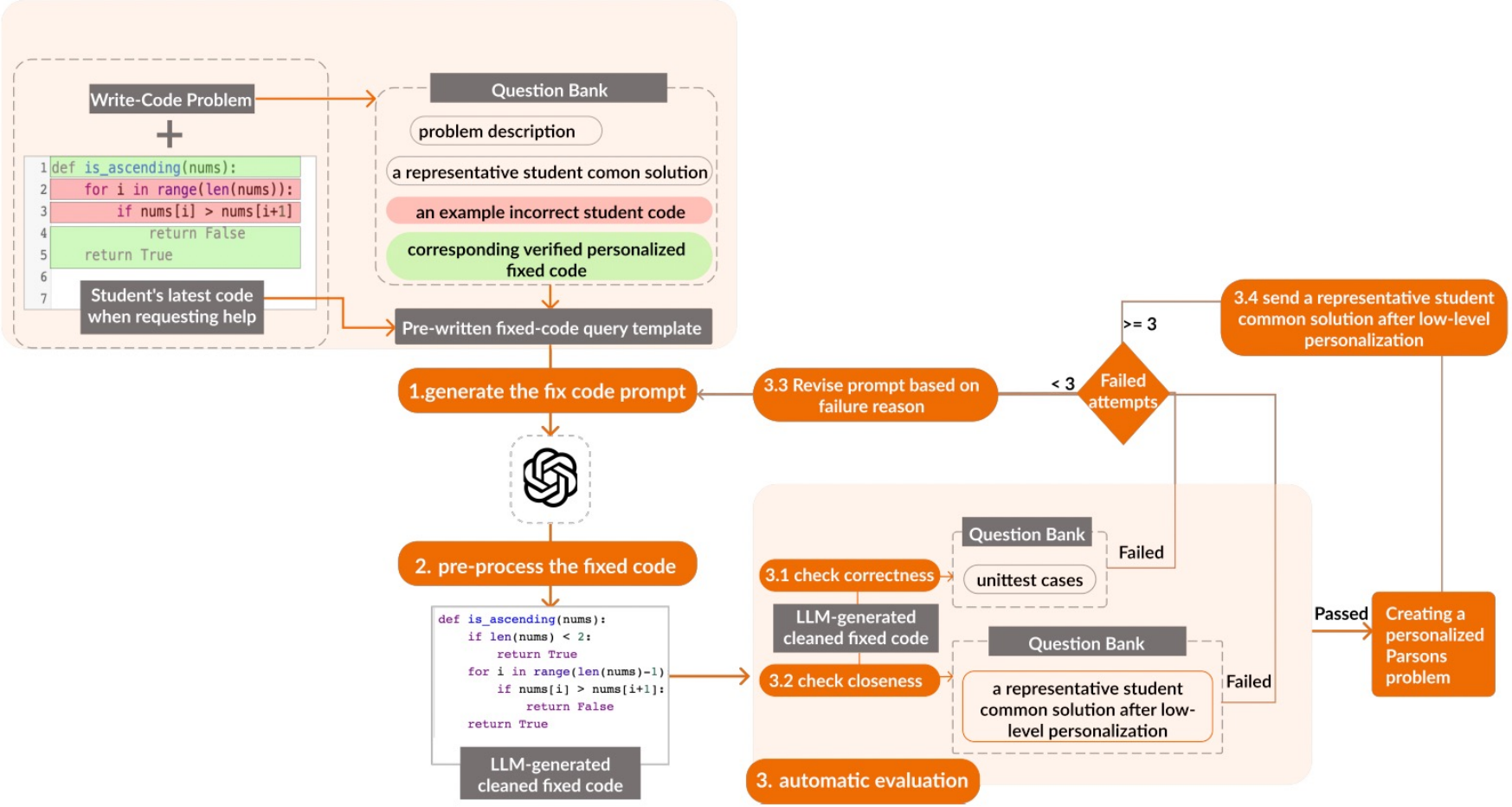


Fig. 5. The ActiveHint's personalized correct solution generation backend

Results

- ActiveHint generates Parsons problems that are significantly similar to the student's incorrect code
  - Vs the most common student solution
- Most students prefer receiving a personalized Parsons problem *vs* just the correct code
  - Some still need an explanation
- Will do an A/B test in Jan 2024
  - Personalized Parsons vs Generated Code

Future Work

- Create fully adaptable Parsons problems
  - Incorporate subgoal labels
    - To help transition from the problem description to code
  - Add different levels of LLM generated explanation
    - High level
    - Statement level
    - Symbol level
- Generate Personalized Micro Parsons for a subgoal
- Try in other fields

Recommendations

- Create tools that leverage LLMs
  - Leverage existing student data
  - Create personalized practice with different levels of explanations
  - Guarantee correct solutions
- Find ways to encourage students to learn
  - Not just copy/paste from LLMs
- Keep students in the Zone of Proximal Development