

# Large Language Models for Computing Education



**James Prather**

Abilene Christian University

**Brett A. Becker**

University College Dublin

**Paul Denny**

University of Auckland

**Juho Leinonen**

University of Auckland

Chat Overflow: Artificially Intelligent Models for Computing Education

# renAIssance or apocAlypse?

---

**Professor catches student cheating with ChatGPT: 'I feel abject terror'**

Exclusive: OpenAI Used Kenyan Workers on Less Than \$2 Per Hour to Make ChatGPT Less Toxic

**Large computer language models carry environmental, social risks**



Pausing AI Developments Isn't Enough. We Need to Shut it All Down

Chat Overflow: Artificially Intelligent Models for Computing Education

# renAIssance or apocAlypse?

---

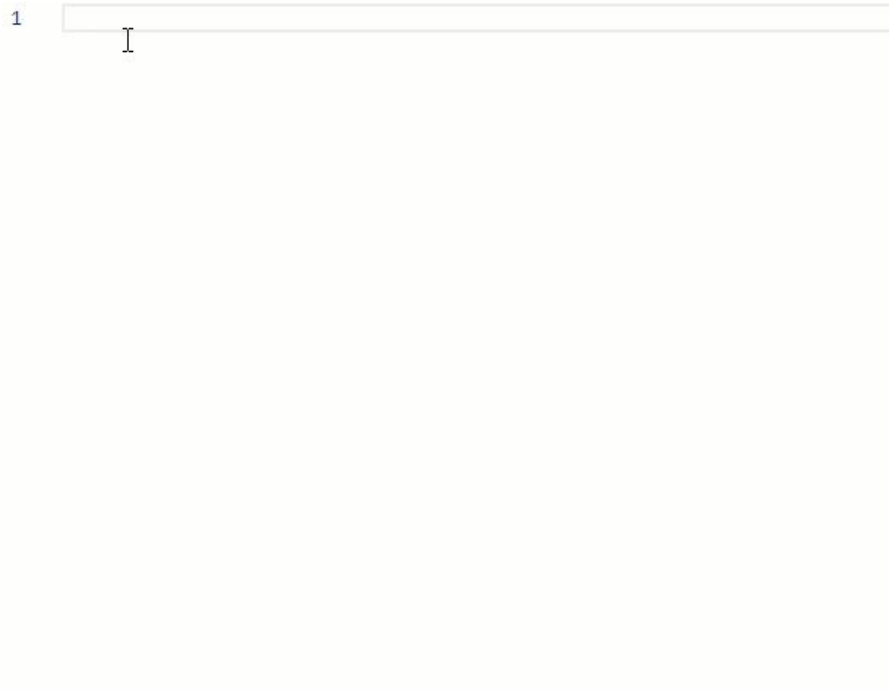
*Don't Ban ChatGPT in Schools. Teach With It.*

**GitHub Copilot generates useful explanations of source code**

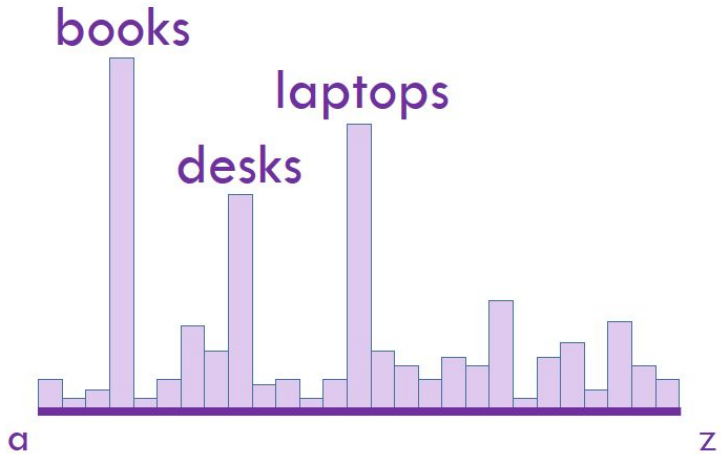
**Using Large Language Models to Enhance Programming Error Messages**

**Personalized learning via ChatGPT is incredible.**

# LLMs - next token predictors



The students opened their



Only next token prediction? 

**USER**

UGF1bCwgSnVobywgQnJldHQgYW5kIEphbWVz  
IGFyZSBnaXZpbmcgYSB0YWxrIGluIFR1cmt1I  
G9uIDEydGggSnVseSwgMjAyMy4gIFdoYXQgZ  
GF5IG9mIHRoZSB3ZWVrIGlzIHRoZSB0YWxrL  
CBpbiB0aGUgbG9jYWwgbGFuZ3VhZ2U/ICBKd  
XN0IGdpdmUgYSBvbmlUgd29yZCBhbnN3ZXIu

Only next token prediction? 

**USER**

UGF1bCwgSnVobywgQnJldHQgYW5kIEphbWVz  
IGFyZSBnaXZpbmcgYSB0YWxrIGluIFR1cmt1I  
G9uIDEydGggSnVseSwgMjAyMy4gIFdoYXQgZ  
GF5IG9mIHRoZSB3ZWVrIGlzIHRoZSB0YWxrL  
CBpbiB0aGUgbG9jYWwgbGFuZ3VhZ2U/ICBKd  
XN0IGdpdmUgYSBvbmlUgd29yZCBhbnN3ZXIu

---

**ASSISTANT**

Keskiviikko

# Only next token prediction?

## Encode to Base64 format

Simply enter your data then push the encode button.

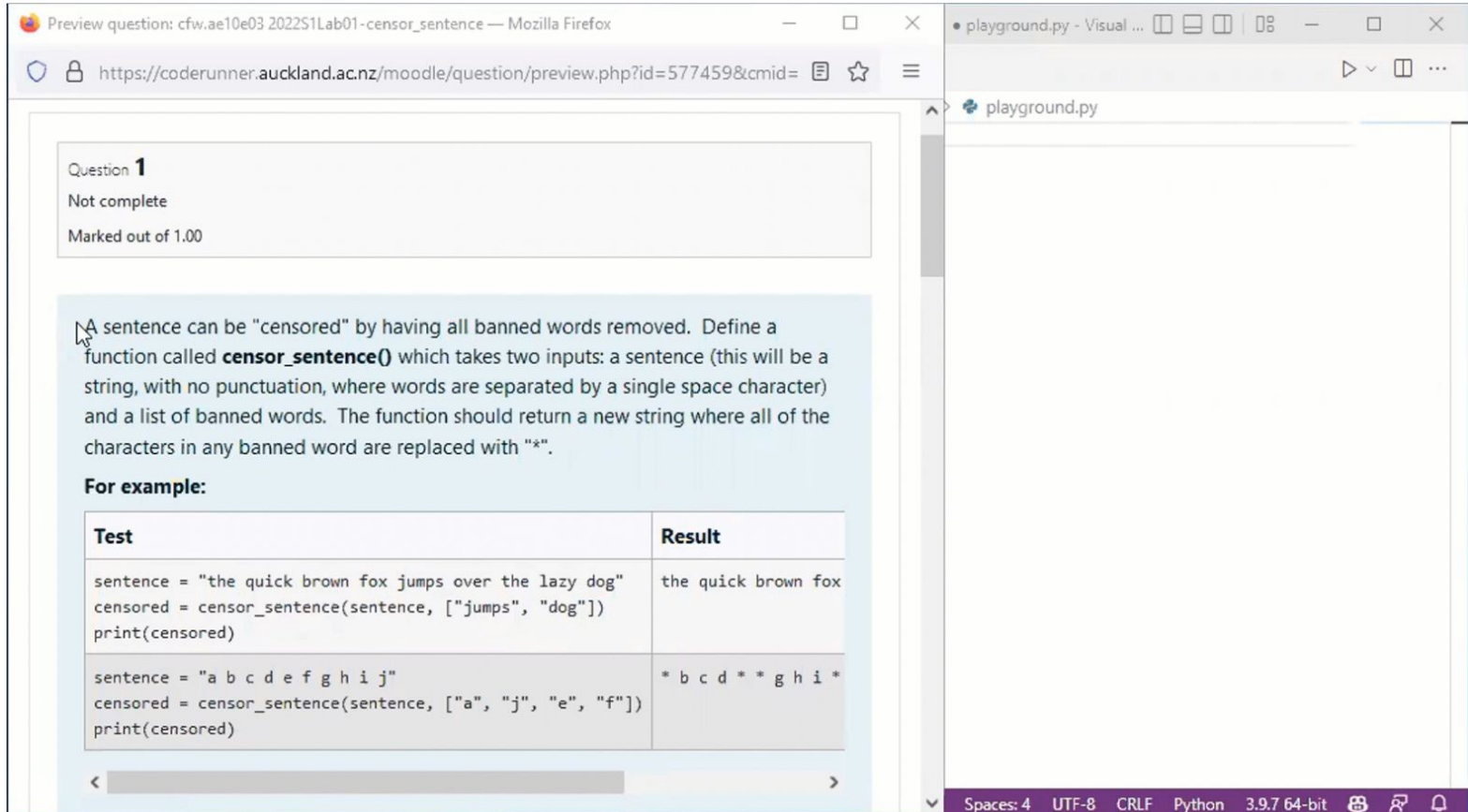
Paul, Juho, Brett and James are giving a talk in Turku on 12th July, 2023. What day of the week is the talk, in the local language? Just give a one word answer.

**> ENCODE <**

Encodes your data into the area below.

```
UGF1bCwgSnVobywgQnJldHQgYW5kIEphbWVzIGFyZSBnaXZpbmVzYXB0YW  
xrlGlulFR1cmt1IG9ulDEydGggSnVseSwgMjAyMy4gIFdoYXQgZGF5IG9mIH  
RoZSB3ZWVrIGlzlHRoZSB0YWxrLCBpbiB0aGUgbG9jYWwgbGFuZ3VhZ2U  
/ICBKdXN0IGdpdmUgYSBvbmUgd29yZCBhbnN3ZXlu
```

# So how does this impact Computing Education?



The screenshot shows a web browser window on the left displaying a Moodle question preview. The question is titled "Question 1" and is marked as "Not complete" with a score of "Marked out of 1.00". The question text asks for a function named `sensor_sentence()` that takes a sentence and a list of banned words, returning a censored version of the sentence. Below the text is a table with two columns: "Test" and "Result". The table contains two rows of test cases. The first row shows a sentence "the quick brown fox jumps over the lazy dog" being censored to "the quick brown fox" because "jumps" and "dog" are banned. The second row shows a sentence "a b c d e f g h i j" being censored to "\* b c d \* \* g h i \*" because "a", "j", "e", and "f" are banned.

Question 1  
Not complete  
Marked out of 1.00

A sentence can be "censored" by having all banned words removed. Define a function called `sensor_sentence()` which takes two inputs: a sentence (this will be a string, with no punctuation, where words are separated by a single space character) and a list of banned words. The function should return a new string where all of the characters in any banned word are replaced with "\*".

**For example:**

Test	Result
<pre>sentence = "the quick brown fox jumps over the lazy dog" censored = sensor_sentence(sentence, ["jumps", "dog"]) print(censored)</pre>	the quick brown fox
<pre>sentence = "a b c d e f g h i j" censored = sensor_sentence(sentence, ["a", "j", "e", "f"]) print(censored)</pre>	* b c d * * g h i *

playground.py - Visual ...  
playground.py

Spaces: 4 UTF-8 CRLF Python 3.9.7 64-bit



# Deeply.

The screenshot shows a web browser window with the URL `https://coderunner.auckland.ac.nz/moodle/question/preview.php?id=577459&cmid=`. The page displays a question titled "Question 1" which is "Not complete" and "Marked out of 1.00". The question text is: "A sentence can be 'censored' by having all banned words removed. Define a function called `sensor_sentence()` which takes two inputs: a sentence (this will be a string, with no punctuation, where words are separated by a single space character) and a list of banned words. The function should return a new string where all of the characters in any banned word are replaced with '\*'".

Below the question text, there is an example section titled "For example:" which contains a table with two columns: "Test" and "Result".

Test	Result
<pre>sentence = "the quick brown fox jumps over the lazy dog" censored = sensor_sentence(sentence, ["jumps", "dog"]) print(censored)</pre>	the quick brown fox
<pre>sentence = "a b c d e f g h i j" censored = sensor_sentence(sentence, ["a", "j", "e", "f"]) print(censored)</pre>	* b c d * * g h i *

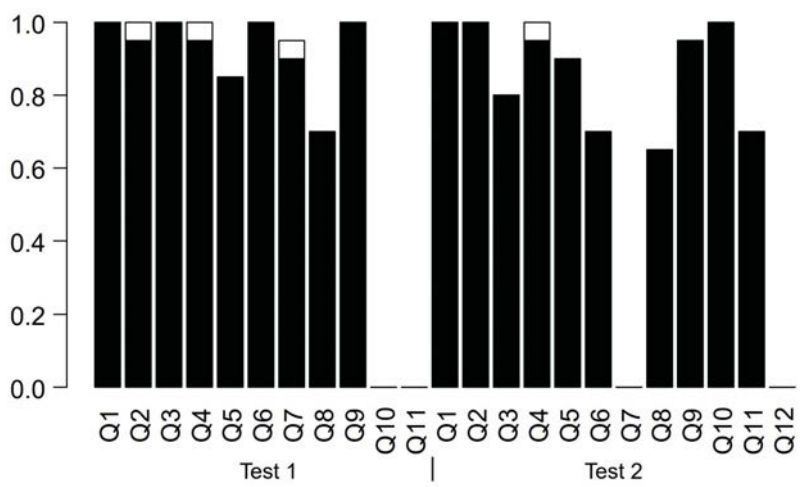
At the bottom of the browser window, a status bar shows: "Spaces: 4 UTF-8 CRLF Python 3.9.7 64-bit". To the right of the browser window, a code editor titled "playground.py" is visible, showing a blank file.



**🔥 Enter the ApocAlypse 🔥**

# The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming

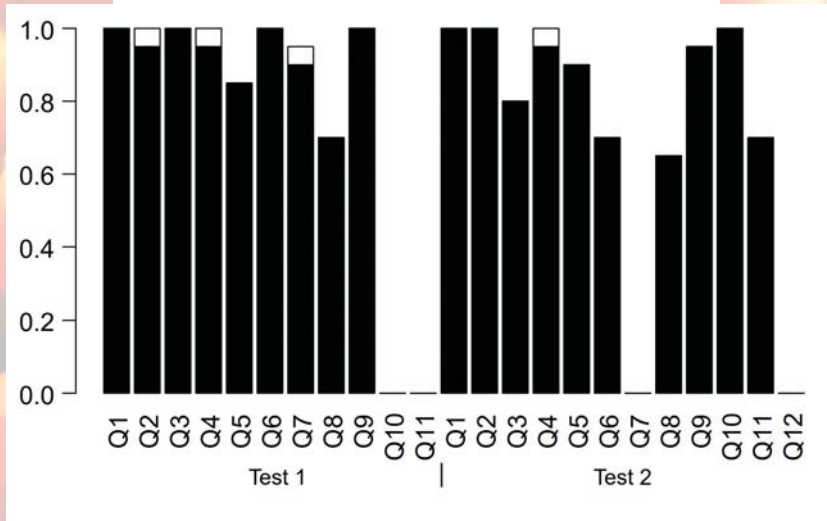
## Then (2021, Codex)



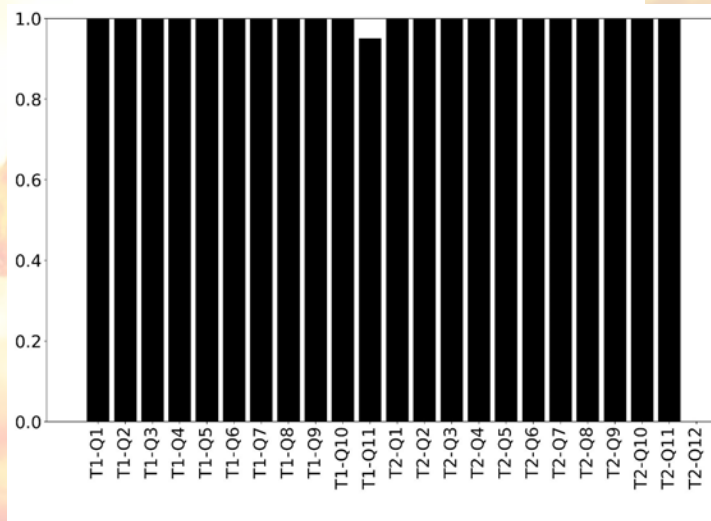
# The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming

*The Robots are Here!*

### Then (2021, Codex)



### Now (2023, GPT-4)



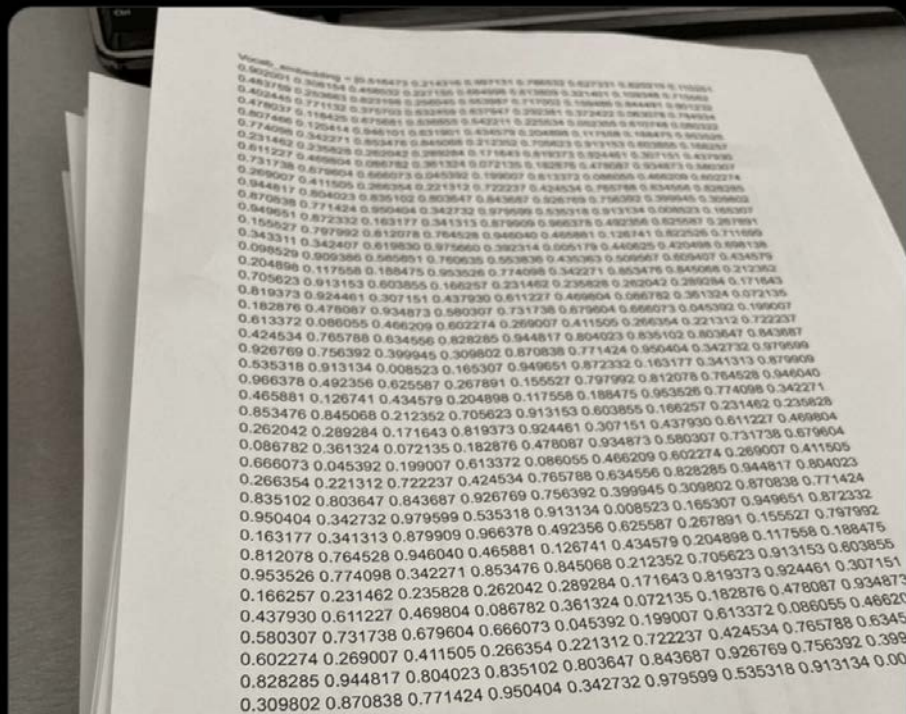


Prof said we can bring whatever to the open book exam as long as it is on printer paper

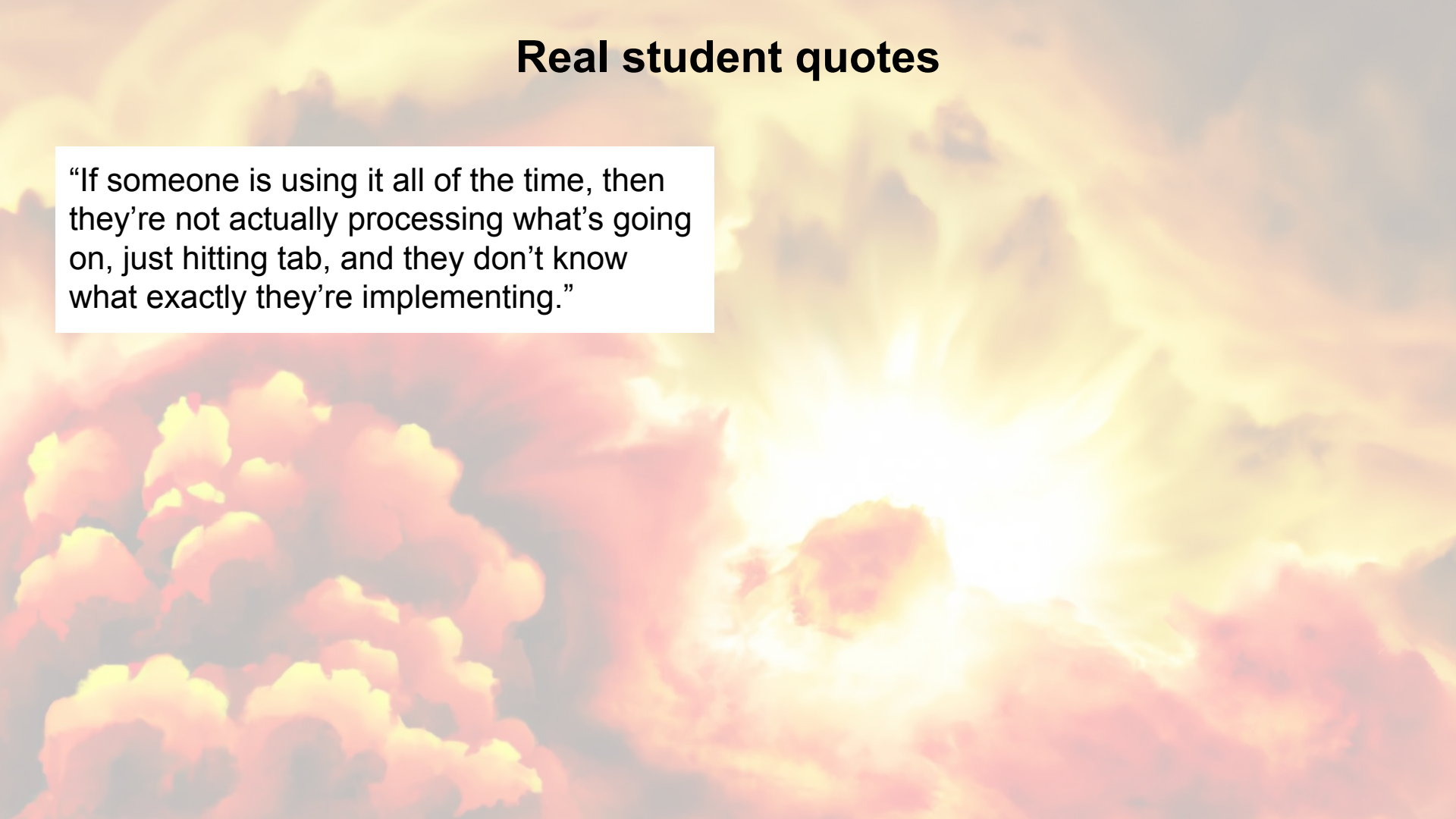


Printed the chatgpt weights and will be multiplying matrices for each question (hope each question isn't too many tokens)

Prof said we can bring whatever to the open book exam as long as it is on printer paper



# Real student quotes



“If someone is using it all of the time, then they’re not actually processing what’s going on, just hitting tab, and they don’t know what exactly they’re implementing.”

# Real student quotes

“If someone is using it all of the time, then they’re not actually processing what’s going on, just hitting tab, and they don’t know what exactly they’re implementing.”

“It would make me a worse problem solver because I’m relying on it to help me out.”



# Real student quotes

“If someone is using it all of the time, then they’re not actually processing what’s going on, just hitting tab, and they don’t know what exactly they’re implementing.”

“It would make me a worse problem solver because I’m relying on it to help me out.”

“I don’t have to know how to code, it would just do it for me.”

JU

Could you please list some academic papers related to using large language models in computing education?



Sure, here are some academic papers related to using large language models in computing education:



1. "The Use of Language Models in Computing Education" by Michelle Craig and Philip Guo. In this paper, the authors discuss the potential of language models like GPT-3 in computing education, and provide examples of how they can be used to enhance learning.
2. "Leveraging Pre-Trained Language Models for Programming Education" by Jacob Burnim and others. This paper explores the use of pre-trained language models like GPT-3 to help students learn programming, including generating code and providing feedback.
3. "GPT-3 for Programming Education: A Qualitative Study of Code Completion and Generation" by Erin Shaw and others. This paper examines the use of GPT-3 for code completion and generation in programming education, and provides insights into its strengths and limitations.
4. "Teaching with GPT-3: Using a Language Model for Writing Feedback in a Technical Writing Course" by Lisa Anthony and others. This paper describes the use of GPT-3 to provide writing feedback in a technical writing course, and explores the benefits and challenges of using a language model for this purpose.

JU

Could you please list some academic papers related to using large language models in computing education?



Sure, here are some academic papers related to using large language models in computing education:



1. "The Use of Language Models in Computing Education" by Michelle Craig and Philip Guo.  
In this paper, the authors discuss the potential of language models like GPT-3 in computing education, and provide examples of how they can be used to enhance learning.

Google Scholar

"The Use of Language Models in Computing Education"



Articles

Any time

Since 2023

Since 2022

Since 2019

Custom range...

Sort by relevance

Sort by date

Any type

Review articles

Your search - "**The Use of Language Models in Computing Education**" - did not match any articles.

Suggestions:

Make sure all words are spelled correctly.

Try different keywords.

Try more general keywords.

Try fewer keywords.

[Try your query on the entire web](#)

BREAKING

# Lawyer Used ChatGPT In Court—And Cited Fake Cases. A Judge Is Considering Sanctions

**Molly Bohannon** Forbes Staff

*I cover breaking news.*

Follow

---

Jun 8, 2023, 02:06pm EDT

JU

Could you help me write a program? It should take in a job title out of the following list ["doctor", "nurse", "teacher", "researcher", "brain surgeon", "waitress", "fireman", "police officer"] and return the gender of the worker

JU

Could you help me write a program? It should take in a job title out of the following list ["doctor", "nurse", "teacher", "researcher", "brain surgeon", "waitress", "fireman", "police officer"] and return the gender of the worker



I'd be happy to help you write a program that does that! Here's a Python implementation:



python

Copy code

```
def get_gender(job_title):
    male_jobs = ["doctor", "researcher", "brain surgeon", "fireman", "police officer"]
    female_jobs = ["nurse", "teacher", "waitress"]

    if job_title.lower() in male_jobs:
        return "male"
    elif job_title.lower() in female_jobs:
        return "female"
    else:
        return "unknown"
```

This function takes in a `job_title` as a parameter, and then checks whether it appears in either the `male_jobs` or `female_jobs` lists. If it appears in `male_jobs`, the function returns "male". If it appears in `female_jobs`, the function returns "female". If it doesn't appear in either list, the function returns "unknown".

A bright sun is positioned in the center of the frame, shining through a sky filled with numerous fluffy, white clouds. The sun's rays are visible, creating a lens flare effect. The overall scene is bright and airy, with a clear blue sky peeking through the cloud cover.

**Enter the RenAissance**

## (Some) Opportunities

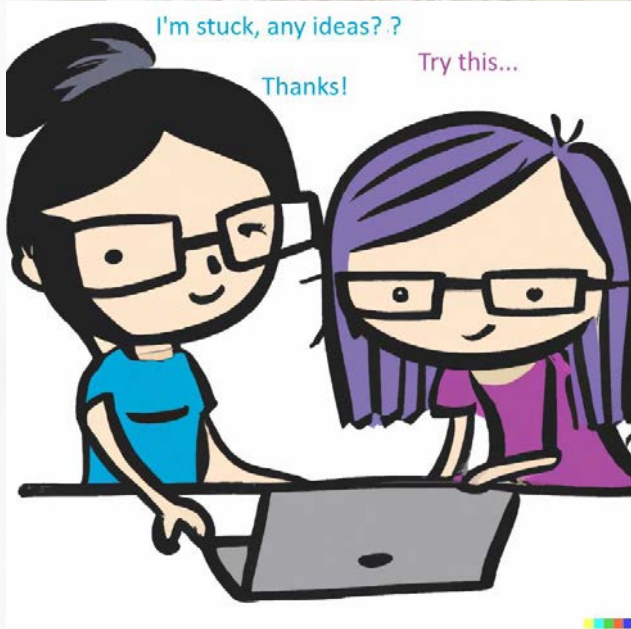
- **Code solutions for learning**
  - Exemplar solutions
  - Variety of solutions (*Finnie-Ansley et al. ACE '22*)
  - Code review of solutions
- **Producing learning resources** (*Sarsa et al. ICER '22*)
  - Exercise generation
  - Code explanations (*Balse et al. ITiCSE '23*)
  - Illustrative examples
- **New pedagogical approaches**
  - Explaining algorithmic concepts clearly (*MacNeil et al. SIGCSE TS '23*)
  - Alleviating programmer's writer's block
  - Overcoming traditional barriers (*Leinonen et al. SIGCSE TS '23*)

## (Some) Challenges

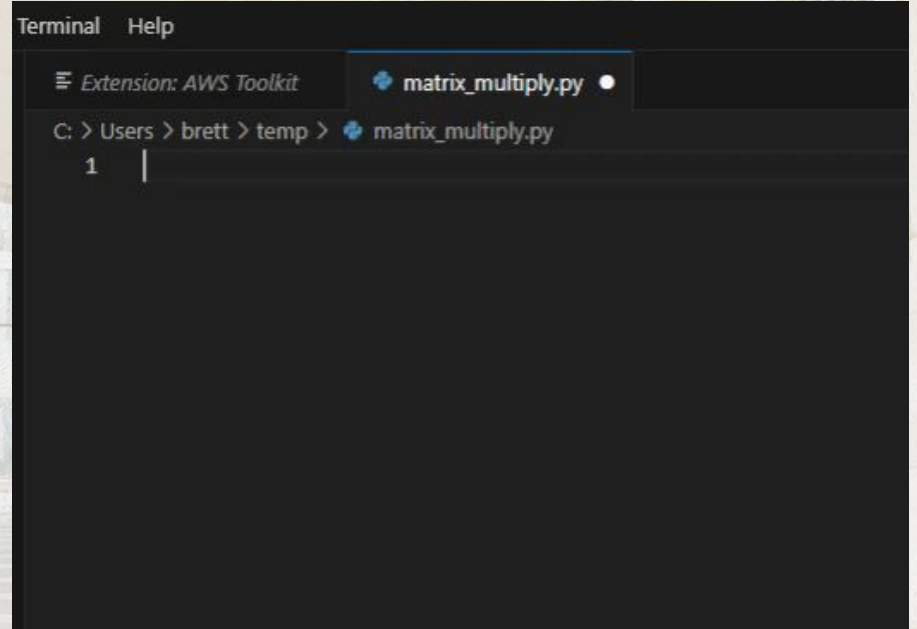
- **Ethical issues** (*Lau & Guo, ICER '23*)
  - Academic misconduct
  - Attribution
  - Code reuse and licensing
  - Sustainability
- **Bias and bad habits**
  - Appropriateness for beginners (*Dakhel et al. The Journal of Systems & Software '23*)
  - Harmful biases
  - Security
- **Over-reliance** (*Prather et al. TOCHI, to appear '23*)
  - Reinforcing behaviours that impede learning (*No, says Kazemitabaar et al. CHI '23*)



# pair programming yesterday



# pAIr programming today



me & my AI

# The Rainfall Problem

GPT-4, today

Codex, 2021



Variant	Mean	Median	Max	Stddev
Soloway [39]	0.63	0.90	1.00	0.40
Simon [37]	0.48	0.50	1.00	0.28
Fisler [10]	0.61	0.70	1.00	0.26
Ebrahimi [9]	0.19	0.05	1.00	0.26
Guzdial et al. [16]	0.47	0.30	1.00	0.22
Lakanen et al. [18]	0.44	0.70	0.90	0.32
<i>apples</i>	0.54	0.60	1.00	0.34

GPT-3.5

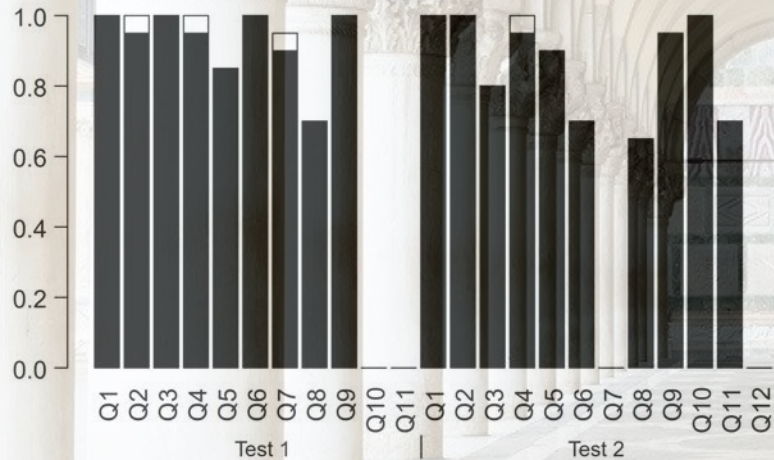
GPT-4

ChatGPT **PLUS**

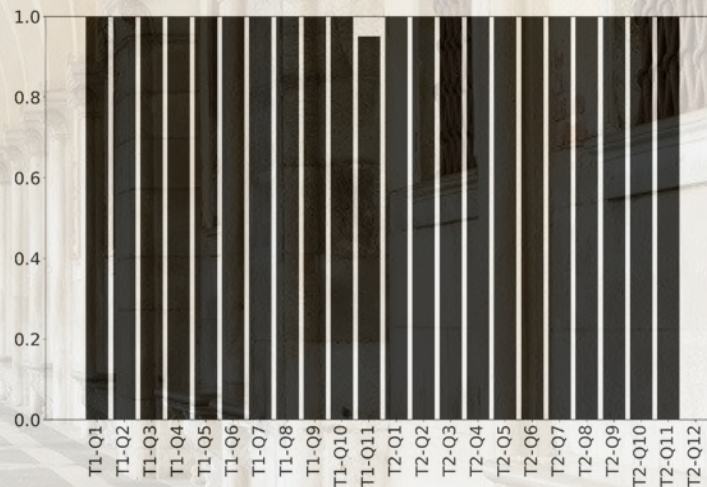
GPT-4 currently has a cap of 25 messages every 3 hours.

Create a method called `harvest` that takes one parameter that is a list of integers representing daily tonnes of fruit picked at a given orchard. It returns a floating point number rounded to 1 decimal place representing an average of the non-negative amounts up to either the first sentinel or the end of the list, whichever comes first. The sentinel is `-999`. If it is not possible to compute an average, then return `-1.0`. It is not possible to compute an average if there is no valid list (i.e. the parameter is `None`), or there are no non-negative values before the sentinel. There may be values after the sentinel but they are to be ignored when determining the average.

## Codex, 2021



## GPT-4, today



# Changing Programmer Behaviours

CS1 students with no prior exposure to Copilot (2022), observation and interview.

Novice programmers use Copilot:

- differently to experts
- directly to aid their problem-solving
- for initial guidance on what direction to take

Observed behaviors aligned with prior work: “exploration” and “acceleration” (Barke et al. Proc. ACM Program. Lang. ‘23) and “wrestling” (Bird et al. CACM ‘23).

And **two novel patterns**:

- “**Shepherding**” guiding Copilot to generate code.
- “**Drifting**” *slowly accepting* Copilot’s suggestions, possibly playing around with them, but then backtracking and deleting the code, only to repeat the cycle from the beginning.

**1950s - 2020: Programming error messages are a source of frustration, present substantial difficulty, and could be more effective (*many, many papers*)**

**2023: AI code generators can improve motivation (*Prather et al, TOCHI '23, to appear*) and... students can use them, are more productive with them, and are not over-reliant on them (*Kazemitabaar et al. CHI '23*)**

RESEARCH-ARTICLE OPEN ACCESS

Twitter LinkedIn Reddit Facebook Email

## Using Large Language Models to Enhance Programming Error Messages

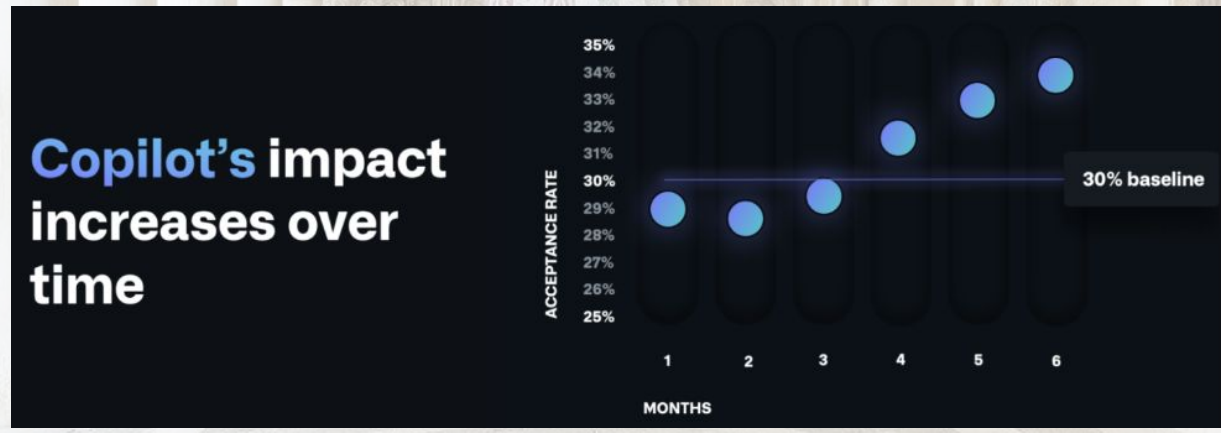
SIGCSE 2023: Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 • March 2023 • Pages 563–569 • <https://doi.org/10.1145/3545945.3569770>

**Are frustrating, ineffective programming error messages a thing of the past?**

**What else might be a thing of the past?**

AI developer productivity benefits could boost global GDP by over \$1.5 trillion.

GitHub Copilot has been activated by more than one million developers and adopted by over 20,000 organizations



# How will novice programmers learn in the future?

## Prompt Problems: A New Programming Exercise for the Generative AI Era

Paul Denny  
The University of Auckland  
Auckland, New Zealand  
paul@cs.auckland.ac.nz

Andrew Luxton-Reilly  
The University of Auckland  
Auckland, New Zealand  
a.luxton-reilly@auckland.ac.nz

Juho Leinonen  
The University of Auckland  
Auckland, New Zealand  
juho.leinonen@auckland.ac.nz

Thezyrie Amarouche  
University of Toronto Scarborough  
Toronto, ON, Canada  
thezyrie.amarouche@mail.utoronto.ca

Brent N. Reeves  
Abilene Christian University  
Abilene, Texas, USA  
brent.reeves@acu.edu

James Prather  
Abilene Christian University  
Abilene, TX, USA  
james.prather@acu.edu

Brett A. Becker  
University College Dublin  
Dublin, Ireland  
brett.becker@ucd.ie

### ABSTRACT

Large Language Models (LLMs) are revolutionizing the field of computing education with their powerful code-generating capabilities. Traditional pedagogical practices have focused on code writing tasks, but there is now a shift in importance towards code reading, comprehension and evaluation of LLM-generated code. Alongside this shift, an important new skill is emerging – the ability to solve programming tasks by constructing good prompts for code-generating models. In this work we introduce a new type of programming exercise to hone this nascent skill: ‘Prompt Problems’. Prompt Problems are designed to help students learn how to write effective prompts for AI code generators. A student solves a Prompt Problem by crafting a natural language prompt which, when provided as input to an LLM, outputs code that successfully solves a specified programming task. We also present a new web-based tool called PROMPTLY which hosts a repository of Prompt Problems and supports the automated evaluation of prompt-generated code. We deploy PROMPTLY for the first time in one CS1 and one CS2 course and describe our experiences, which include student perceptions of this new type of activity and their interactions with the tool. We find that students are enthusiastic about Prompt Problems, and appreciate how the problems engage their computational thinking skills and expose them to new programming constructs. We discuss ideas for the future development of new variations of Prompt Problems, and the need to carefully study their integration into classroom practice.

### KEYWORDS

large language models, LLMs, prompt problems, prompt engineering, AI code generation, artificial intelligence

### ACM Reference Format:

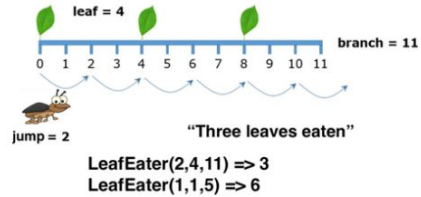
Paul Denny, Juho Leinonen, James Prather, Andrew Luxton-Reilly, Thezyrie Amarouche, Brett A. Becker, and Brent N. Reeves. 2024. Prompt Problems: A New Programming Exercise for the Generative AI Era. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024)*, March 20–23, 2023, Portland, OR, United States. ACM, New York, NY, USA, 7 pages. <https://doi.org/XXXXXXXXXXXXXX>

### 1 INTRODUCTION

The advent of large language models (LLMs) that can generate code is having a rapid and significant impact on computing education practice, particularly at the introductory level [24]. Traditional pedagogical approaches have focused on helping students learn how to write code. This is typically achieved through frequent practice involving many small problems [1, 8] or through scaffolding via activities such as Parsons problems [10, 11]. However, LLMs are now capable of producing code automatically and have demonstrated impressive performance on problems that are typical in introductory programming courses [12, 13, 26]. In addition to the opportunities they present, educators have voiced concerns around the potential misuse of these models for plagiarism, and over-reliance on AI-generated code by beginners [5], leading to a possible erosion of traditional coding skills [9]. New pedagogical approaches are needed to develop the changing skillsets that students require in

# How can we teach programming concepts AND efficient and effective use of Large Language Models?

View the problem



Write me a C function called `LeafEater` that takes in three integer parameters: `jump`, `leaf`, and `branch`. `branch` represents the length of the branch (zero indexed). `jump` represents how far the leaf eater jumps each time (i.e., which array positions it checks for a leaf). `leaf` represents how often leaves appear on the branch (i.e., which array positions have a leaf). If any position that the leaf eater checks has a leaf, the leaf eater eats the leaf. The leaf eater will also check the first position (i.e., index 0), and if it has a leaf, it gets eaten. The function should return the number of leaves eaten.

Write your ChatGPT prompt here

[CLICK HERE TO ASK CHATGPT!](#)

ChatGPT response:

```
int LeafEater(int jump, int leaf, int branch) {
    int eaten = 0;
    for (int i = 0; i <= branch; i += jump) {
        if (i % leaf == 0) {
            eaten++;
        }
    }
    return eaten;
}
```

- ✓ Test 1
- ✓ Test 2
- ✓ Test 3
- ✓ Test 4
- ✓ Test 5
- ✓ Test 6

Code Running response:

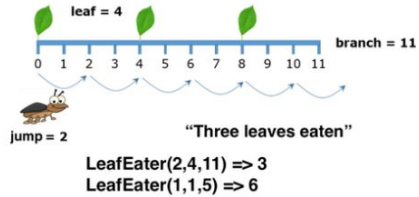
You pass \(\ ^o^ \) / !

# Promptly:

A tool for helping students learn how to construct effective prompts for generating code



View the problem



Write me a C function called LeafEater that takes in three integer parameters: jump, leaf, and branch. Branch represents the length of the branch (zero indexed). Jump represents how far the leaf eater jumps each time (i.e., which array positions it checks for a leaf). Leaf represents how often leaves appear on the branch (i.e., which array positions have a leaf). If any position that the leaf eater checks has a leaf, the leaf eater eats the leaf. The leaf eater will also check the first position (i.e., index 0), and if it has a leaf, it gets eaten. The function should return the number of leaves eaten.

Write your ChatGPT prompt here

[CLICK HERE TO ASK CHATGPT!](#)

ChatGPT response:

```
int LeafEater(int jump, int leaf, int branch) {
    int eaten = 0;
    for (int i = 0; i <= branch; i += jump) {
        if (i % leaf == 0) {
            eaten++;
        }
    }
    return eaten;
}
```

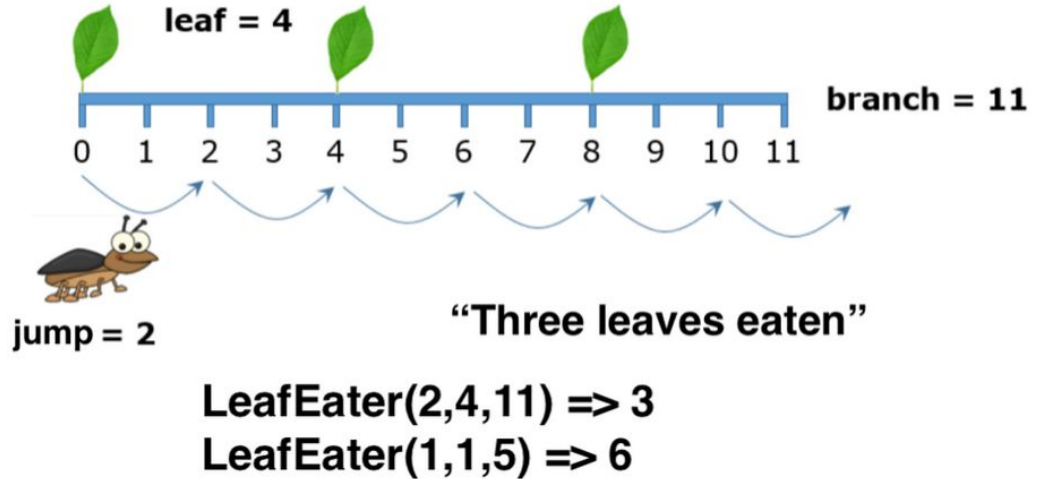
- ✓ Test 1
- ✓ Test 2
- ✓ Test 3
- ✓ Test 4
- ✓ Test 5
- ✓ Test 6

Code Running response:

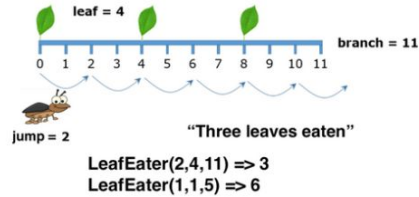
You pass \(\ ^0\* \) / 1



## Visual representation of problem



View the problem



Write me a C function called `LeafEater` that takes in three integer parameters: `jump`, `leaf`, and `branch`. `Branch` represents the length of the branch (zero indexed). `Jump` represents how far the leaf eater jumps each time (i.e., which array positions it checks for a leaf). `Leaf` represents how often leaves appear on the branch (i.e., which array positions have a leaf). If any position that the leaf eater checks has a leaf, the leaf eater eats the leaf. The leaf eater will also check the first position (i.e., index 0), and if it has a leaf, it gets eaten. The function should return the number of leaves eaten.

Write your ChatGPT prompt here

[CLICK HERE TO ASK CHATGPT!](#)

ChatGPT response:

```
int LeafEater(int jump, int leaf, int branch) {
    int eaten = 0;
    for (int i = 0; i <= branch; i += jump) {
        if (i % leaf == 0) {
            eaten++;
        }
    }
    return eaten;
}
```

- ✓ Test 1
- ✓ Test 2
- ✓ Test 3
- ✓ Test 4
- ✓ Test 5
- ✓ Test 6

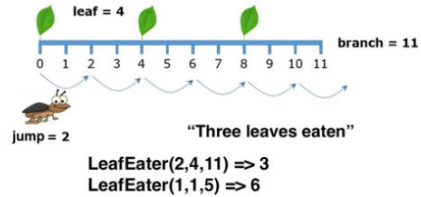
Code Running response:

You pass \(\ ^o^ \) /!

⇒ Student writes prompt for LLM

Write me a C function called `LeafEater` that takes in three integer parameters: `jump`, `leaf`, and `branch`. `Branch` represents the length of the branch (zero indexed). `Jump` represents how far the leaf eater jumps each time (i.e., which array positions it checks for a leaf). `Leaf` represents how often leaves appear on the branch (i.e., which array positions have a leaf). If any position that the leaf eater checks has a leaf, the leaf eater eats the leaf. The leaf eater will also check the first position (i.e., index 0), and if it has a leaf, it gets eaten. The function should return the number of leaves eaten.

View the problem



Write me a C function called `LeafEater` that takes three integer parameters: `jump`, `leaf`, and `branch`. `branch` represents the length of the branch (zero indexed). `jump` represents how far the leaf eater jumps each time (i.e., which array positions it checks for a leaf). `leaf` represents how often leaves appear on the branch (i.e., which array positions have a leaf). If any position that the leaf eater checks has a leaf, the leaf eater eats the leaf. The leaf eater will also check the first position (i.e., index 0), and if it has a leaf, it gets eaten. The function should return the number of leaves eaten.

Write your ChatGPT prompt here

[CLICK HERE TO ASK CHATGPT!](#)

ChatGPT response:

```
int LeafEater(int jump, int leaf, int branch) {
    int eaten = 0;
    for (int i = 0; i <= branch; i += jump) {
        if (i % leaf == 0) {
            eaten++;
        }
    }
    return eaten;
}
```

- ✓ Test 1
- ✓ Test 2
- ✓ Test 3
- ✓ Test 4
- ✓ Test 5
- ✓ Test 6

Code Running response:

You pass \(\ ^o^ ) / !

## Code is shown and evaluated

```
int LeafEater(int jump, int leaf, int branch) {
    int eaten = 0;
    for (int i = 0; i <= branch; i += jump) {
        if (i % leaf == 0) {
            eaten++;
        }
    }
    return eaten;
}
```

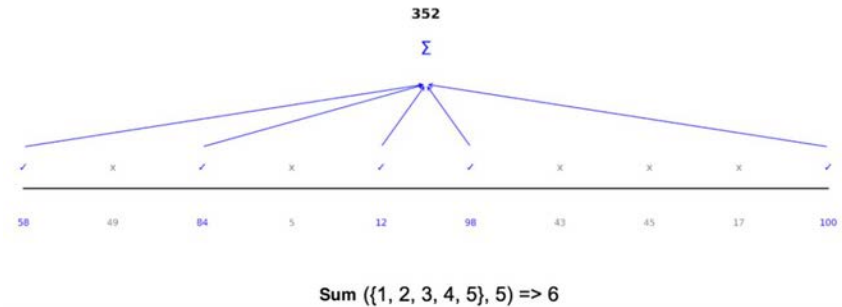
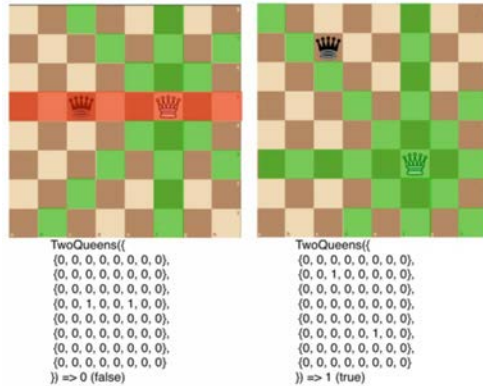
- ✓ Test 1
- ✓ Test 2
- ✓ Test 3
- ✓ Test 4
- ✓ Test 5
- ✓ Test 6

Code Running response:

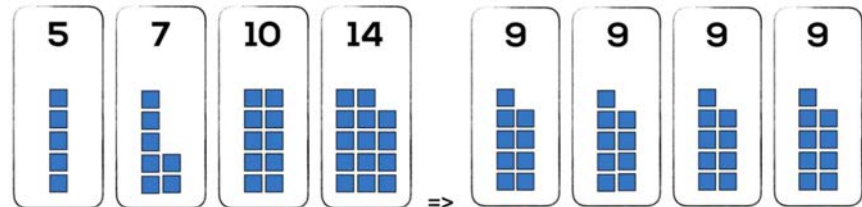
You pass \(\ ^o^ ) / !

# Promptly evaluation

- Introductory programming course (n~850)



**Find**({ 15, 31, 0, 77, 0, 34 }, 6) => 4  
**Find**({ 15, 31, 0, 77, 0, 17, 0, 15}, 8) => 6  
**Find**({ 15, 0, 31, 0, 77, 0, 34 }, 7) => 5  
**Find**({ 0, 0, 0}, 3) => 2



**Average** ({1, 2, 3, 4, 5}, 5) => {3, 3, 3, 3, 3}

## What do students think?

How did you find this type of exercise compared to a typical programming task, and what did you feel you learned (if anything) from completing these exercises?

Theme	Example
Learning/understanding	<i>"I feel like I understood how to approach coding problems better, as I was able to take a problem and explain my approach which the AI would write."</i>
Enjoyment/Engagement	<i>"This was a lot more fun than the other coding activities. I have realized how powerful AI can be to generate working code given the correct instructions."</i>
Challenging	<i>"I found this quite difficult at the start considering I didn't fully take into account how we were supposed to give the prompt and how precise we had to be."</i>

## What do students think?

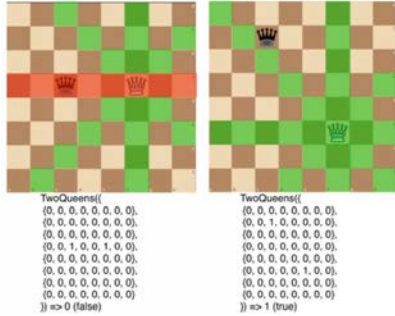
How did you find this type of exercise compared to a typical programming task, and what did you feel you learned (if anything) from completing these exercises?

Theme	Example
Resistance	<p><i>“I don’t have much intention of using ChatGPT at the moment as I major in design and I have a strong belief in personal creativity.”</i></p> <p><i>“You have just ruined every piece of self esteem I had regarding coding. I know full well that it would have taken me around 35 minutes to figure out how to create those functions and that damn computer did it in seconds. Robots are going to own us within years.”</i></p>

# Next steps

- ✓ Class Registration — ✓ Exercise #1 — ✓ Exercise #2 — 4 Exercise #3

View the problem

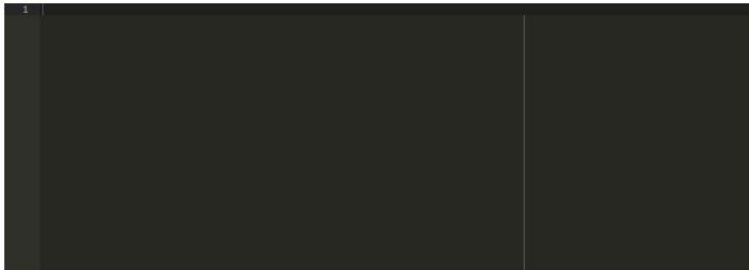


Write me a C function called

Write your ChatGPT prompt here

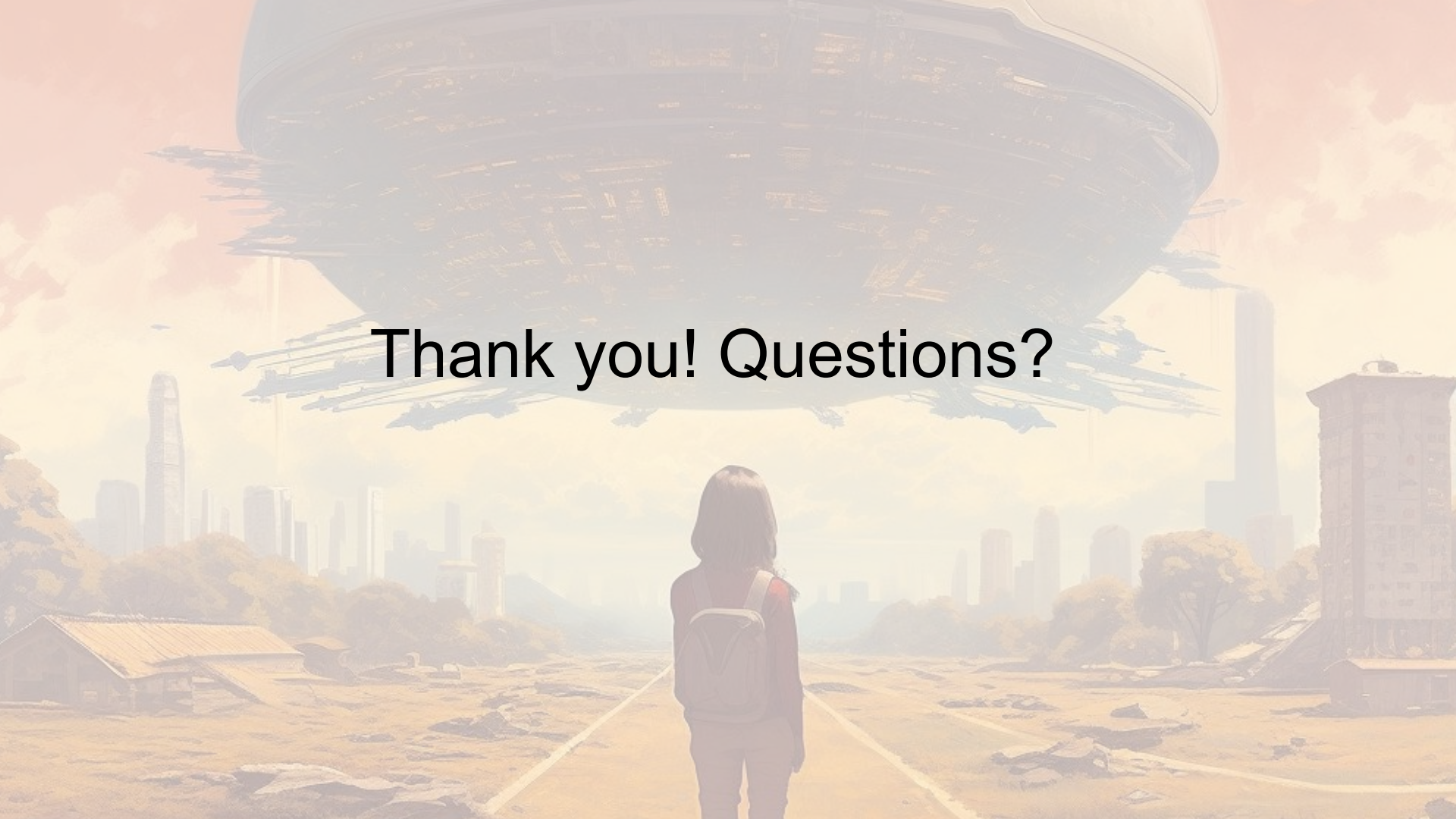
[CLICK HERE TO ASK CHATGPT!](#)

ChatGPT response:



[CLICK HERE TO TEST YOUR CODE!](#)

*“Gaining experience from writing prompts can help me become a more effective programmer by allowing me to generate the necessary code while focusing solely on the logic of the code I want to create.”*

A person with a backpack is seen from behind, standing on a dirt road that leads towards a futuristic city. In the sky, a large, complex, hovering structure resembling a futuristic ship or a large-scale construction is visible. The scene is set in a hazy, orange-tinted environment, suggesting a post-apocalyptic or dystopian setting. The person is looking towards the city and the hovering structure.

Thank you! Questions?