World Journal of Information Systems 2(1) 1-9
DOI: https://doi.org/10.17013/wjis.v2i1.27

Article

Enhancing Introductory Coding Courses with ChatGPT

Anna L. McNab [1,*] and Christopher P. Wright [1]

[1] Department of Management, Niagara University – Lewiston, New York, U.S.A.

***** Correspondence: amcnab@niagara.edu

(Article history: Received December 18, 2024; Received in revised form January 25, 2025; Accepted February 24, 2025; Available online March 15, 2025)

Abstract: This paper explores the integration of generative-AI-powered tools like ChatGPT into teaching introductory coding courses, focusing specifically on a Python course at the college level. It discusses the potential of these tools to revolutionize the learning experience by providing interactive, personalized, and immediate support to students, much like a personal digital tutor. The paper outlines five types of prompts where ChatGPT can be particularly beneficial: (1) clarifications of coding concepts, (2) additional explanations for challenging concepts, (3) coding exercises with problem statements and feedback, (4) personalized learning experiences with adaptive challenges, and (5) project guidance. It highlights the advantages of using ChatGPT and addresses potential challenges in its integration.

Keywords: Generative AI in Education; Python; ChatGPT

1. Introduction

The meteoric rise of generative artificial intelligence (AI) tools such as ChatGPT over the past couple of years has sparked intense conversation about its use in education. Although studies remain limited, researchers have begun to investigate these applications. One of the most comprehensive studies to date completed a content analysis of approximately one hundred news articles and found that most of the early discussion focuses on academic integrity concerns, though others are beginning to discuss ChatGPT and similar tools as an opportunity to design innovative assignments and assessments [1]. Authors have also examined how fake citations are produced [2], how AI can support essay writing [3], and have encouraged critical reflection on AI's use in society [4]. Furthermore, in the context of AI-powered chat technology like ChatGPT, [5] found that students are comfortable adopting new technologies and that frequent use fosters habitual behavior.

Though most continue to have very mixed feelings about implementing artificial intelligence into education, there is no doubt the technology will permeate all aspects of our lives. As such, it seems logical to look for ways in which using the technology effectively

© 2025 ITMA. All rights reserved.

might be beneficial. There are many different AI-powered tools that can be used and new ones are being created daily. The following paper will discuss how to implement ChatGPT into teaching introductory coding courses, in this case a Python course at the college level. Early research shows somewhat mixed sentiment when it comes to implementing ChatGPT into teaching coding. In a study done by [6], a Python code generated by ChatGPT was compared with code generated by students. The authors conclude that ChatGPT was performing as mid-range students, but did not outperform better students, and they did not recommend using it in the future. While these findings shall be respected, perhaps guiding students towards collaborating with ChatGPT, rather than replacing their work altogether, may be particularly beneficial.

Building on prior work, this paper proposes five main ways in which ChatGPT can be used to improve the learning experience for our students by offering interactive, personalized, and immediate support when used in addition to the teaching offered by the professor. This approach can complement traditional teaching methods, providing a blend of instruction that caters to diverse learning styles and needs and can be especially helpful to students who require additional help. It this way, it acts much like a personal, digital tutor.

2. Five Types of Use with an Example Prompt for ChatGPT

The following section provides additional detail for each of the five different uses with a sample prompt for ChatGPT, as well as the code output where available. The idea is to showcase what is possible, not to prescribe these exercises as the only ones that can be used. Due to the nature of generative AI, it is important to note that repeating these same prompts will not necessarily create identical output.

2.1. Clarification

Clarification of concepts can be especially helpful for beginner students who may be confused with syntax. The example provided in Table 1 shows a student asking for the difference between two different commands. ChatGPT not only provides the definition, but also provides an example of use, further clarifying the concepts, similar to what a Professor might do in class or a tutor might do in a tutoring session.

Table 1. Clarification Example

Student Prompt	What's the difference between `==` and `is` in Python?
ChatGPT Response	In Python, `==` is used to compare the value or equality of two objects, whereas `is` is used to compare the identities of two objects, meaning it checks whether both objects refer to the same memory location. For example: **a = [1, 2, 3]** **b = [1, 2, 3]**

© 2025 ITMA. All rights reserved.

	print(a = = b) # True, because the contents are the same **print(a is b)** # False, because they are different objects in memory

2.2. Additional Explanation

Chat GPT can also offer additional explanation for concepts the student may be struggling with. Students can also ask ChatGPT to clarify doubts on coding concepts, syntax, or logic at any time, receiving explanations that help them overcome obstacles without having to wait for the next class or office hours. An example showing how a student may ask for additional explanation for a concept they struggle with is provided in Table 2 below.

Table 2. Additional Explanation Example

Student Prompt	I'm struggling with recursion in Python. Can you explain it with an easy example?
ChatGPT Response	Of course! Think of recursion as a way for a function to call itself to solve a problem. A simple example is calculating the factorial of a number. The factorial of a number n (denoted as n!) is the product of all positive integers less than or equal to n. Here's a simple recursive function for it: **def factorial (n):** **if n = = 1:** **return 1** **else:** **return n * factorial (n-1)** This function calls itself, reducing the problem size each time, until it reaches the base case (n = = 1).

2.3. Coding Exercises

Students can use ChatGPT to practice coding exercises where the AI provides a problem statement for students to practice their skills. The beauty of generative AI is that it will generate a different response every time a student asks for an example. One such example is provided in Table 3 below.

Table 3. Coding Exercises Example

Student Prompt	Can you give me a simple Python exercise to practice loops?
ChatGPT Response	Sure! Here's a basic exercise: Write a Python program that asks the user for a number and then prints out a list of all the numbers from 1 to that number that are divisible by 3. Remember to use a loop to iterate through the numbers.

As students code their solution, if desired, in the next step ChatGPT can offer hints, suggestions, and immediate feedback on errors or bugs in their code. An example of this type of interaction is provided later.

© 2025 ITMA. All rights reserved.

2.4. Personal Learning

ChatGPT can adapt to each student's learning pace, offering simpler or more complex coding challenges based on their proficiency. This ensures that students are neither overwhelmed nor bored, keeping them engaged and motivated. This can be an incredible aid for professors who do not have the bandwidth to offer a personalized problem for every student. Additionally, ChatGPT can assist with intermediary steps offering further support.

An example of personalized learning is provided in Table 4. This is an example of a conversational interaction, in which the student receives the first response from ChatGPT, follows it with another prompt, and then receives an additional response back. This also highlights that ChatGPT "understands" what the student is asking about and is available to explain the concept to the student immediately.

Table 4. Personal Learning Example

Student Prompt	Here's a simple recursive function in python that I understand: **def factorial(n):** **if n == 1:** **return 1** **else:** **return n * factorial(n-1)** can you provide instructions for a slightly more complex problem practicing the same skills? - do not provide the solution, just the problem statement. I will try to solve it and then later might ask you for the solution.
ChatGPT Response	Certainly! Here's a problem statement for a slightly more complex problem that requires similar recursive skills: **Problem Statement: Recursive Function to Calculate the Sum of Digits** **Objective:** Write a recursive function in Python that calculates the sum of all the digits of a positive integer. The function should take a single integer as input and return the sum of its digits. **Requirements:** – If the input is a single digit, the function should return that digit. – For a number greater than 9, the function should return the sum of its digits. For example, if the input is `123`, the function should return `6` (which is `1 + 2 + 3`). – The function must use recursion to solve the problem; you cannot use loops or string conversions. **Hints:** – Think about how you can reduce the problem for the recursive call. How does splitting off the last digit of the number help you move toward the base case? – Consider the mathematical operations that might help you isolate digits of the number.

© 2025 ITMA. All rights reserved.

	Feel free to work on this problem and ask for hints or the solution if needed!
Additional Student Prompt Asking for Help Troubleshooting Student Solution	I am really struggling with figuring out how to define the recursive case. Can you help me with the step.
ChatGPT Response	Certainly! The key to solving this problem is understanding how to break down the recursive case. Let's dive into that. **Understanding the Recursive Case** For a number greater than 9, you need to find a way to sum its digits recursively. This involves two steps: 1. **Isolating the last digit:** You can get the last digit of a number `n` by using the modulo operation `n % 10`. 2. **Reducing the number:** You can remove the last digit from `n` by dividing `n` by 10 and taking the floor of that number, which is effectively done by the integer division operation `n // 10`. Now, to define the recursive case, you will: – **Sum the last digit** (found using `n % 10`) with – **A recursive call** to the same function with the reduced number (`n // 10`) as its argument. This means your function will keep calling itself, each time with a smaller version of the original number, until it reaches a single digit. At that point, the base case takes over and the recursion starts unwinding, summing up all the last digits found during each call. **Implementing the Recursive Case** Here's how you can put this understanding into code: **def sum_of_digits(n):** **# Base case: if n is a single digit, simply return it.** **if n < 10:** **return n** **else:** **# Recursive case: return the last digit plus the sum of the digits of the rest of the number.** **return (n % 10) + sum_of_digits(n // 10)**

© 2025 ITMA. All rights reserved.

	How It Works Let's walk through an example with `n = 123` to see the function in action: – First call: `n = 123` – Last digit: `123 % 10 = 3` – Rest of the number: `123 // 10 = 12` – Returns: `3 + sum_of_digits(12)` – Second call: `n = 12` – Last digit: `12 % 10 = 2` – Rest of the number: `12 // 10 = 1` – Returns: `2 + sum_of_digits(1)` – Third call: `n = 1` (Base case) – Returns: `1` So, the total return value is `3 + 2 + 1 = 6`, as expected. This step-by-step approach demonstrates how recursion breaks down the problem into manageable chunks, each step simplifying the problem until it reaches the base case.

2.5. Project Guidance

Finally, for coding projects, students can consult ChatGPT on design decisions, best practices, and debugging tips, making the project development process more efficient and educational. Of course, it does not prevent the student from having ChatGPT work out a solution to the project problem. A discussion of permitted and prohibited uses of ChatGPT is recommended to ensure students understand the rules of conduct before starting a project. The example shown in Table 5 provides a concise guide for a student beginning a web scraping project, outlining the steps and tools required, and illustrating how to implement them with a simple Python script.

Table 5. Project Guidance Example

Student Prompt	I'm trying to build a simple web scraper in Python for my project, but I'm not sure where to start. Can you guide me through the basic steps?
ChatGPT Response	Absolutely, I'd be happy to help you get started on your web scraping project. Web scraping involves extracting data from websites, and Python provides several libraries to make this task easier. One popular choice is Beautiful Soup. Here's a high-level overview of the steps you'll follow: 1. **Identify the URL of the website you want to scrape**. Make sure you have the legal right to scrape the website and that it complies with the site's `robots.txt` file and terms of service. **2. Install Beautiful Soup and Requests.** You'll need these libraries, so if you haven't installed them yet, you can do so by running `pip install beautifulsoup4 requests` in your terminal.

© 2025 ITMA. All rights reserved.

	3. Fetch the webpage with Requests.
	Use the Requests library to make an HTTP request to the website's URL and fetch the HTML content of the page.
	4. Parse the HTML content with Beautiful Soup.
	Pass the HTML content to a Beautiful Soup object. This step allows you to navigate and search through the HTML for the data you're interested in.
	5. Extract the information.
	Use Beautiful Soup's methods to find the HTML elements containing the data you want to scrape. You can search for elements by tags, classes, or IDs.
	
	Here's a simple example that scrapes the titles from the Python Wikipedia page:
	
	import requests
	from bs4 import BeautifulSoup
	
	# Step 1: URL to scrape
	url = 'https://en.wikipedia.org/wiki/Python_(programming_language)'
	# Step 2: Fetch the content of the webpage
	response = requests.get(url)
	# Step 3: Parse the HTML content
	soup = BeautifulSoup(response.text, 'html.parser')
	# Step 4: Extract titles of sections for heading in soup.find_all('span', class_='mw-headline'):
	print(heading.text)
	
	This script will print out the titles of different sections on the Python Wikipedia page. Remember, this is just a starting point. Your project may require more complex data extraction, so you'll need to adapt the code to fit your specific needs. Always ensure that you're respecting the website's scraping policies.

3. Benefits Of Adding ChatGPT to Traditional Methods

Before we address the obvious hesitation most faculty have with allowing ChatGPT in their classrooms and their concern the students will simply ask it for solutions (which they will whether we like it or not) we will discuss some benefits of actually encouraging students to use ChatGPT while learning to code.

3.1. Immediate Feedback, Support and Availability:

Unlike traditional settings where feedback can be delayed, ChatGPT provides instant feedback on coding assignments and exercises. This immediate reinforcement aids in faster

© 2025 ITMA. All rights reserved.

learning and correction of mistakes, allowing students to progress more quickly. Furthermore, ChatGPT is accessible at any time, making it an invaluable resource for students working on coding exercises outside of regular class hours. This can significantly enhance learning, especially for students in different time zones or with varying schedules.

3.2. Customized Learning Experiences:

Traditional teaching methods in a conventional classroom setting must often follow a one-size-fits-all approach, leaving the star students lacking challenge and the weaker students struggling to keep up. In contrast, ChatGPT can offer a more personalized learning experience adapted to the student's pace. Furthermore, ChatGPT can make learning to code more interactive and engaging through conversation-based exercises and instant feedback. This interaction can increase student interest and motivation, leading to better learning outcomes for both the students who were seeking more challenge and those who needed additional help to keep up.

3.3. Supplemental & Autonomous Learning:

Much like a personal tutor, ChatGPT acts as a supplement to traditional lectures and textbooks, offering an additional resource for students to explore concepts and practice coding. This blended learning approach can cater to various learning preferences, enhancing the overall educational experience. Additionally, because of its 24-7 availability, students are able to take complete charge of their own learning, exploring topics and solving problems independently. This autonomy in learning encourages the development of critical thinking and problem-solving abilities, key competencies in programming and beyond. Such skills are vital for continued success long after graduation.

4. Implementation Considerations

While integrating ChatGPT into introductory coding courses offers numerous benefits, educators should also consider potential challenges. Ensuring that interactions with ChatGPT are structured to prevent over-reliance on the tool for solutions is crucial. It is also important that ChatGPT is implemented in a way that *complements*, rather than *replaces*, traditional instructional methods, preserving the invaluable human element of teaching.

Educators should provide guidance on how to effectively use ChatGPT, emphasizing its role as a learning aid rather than a source of answers. Initially, educators may need to provide students with guidelines on how to construct effective prompts for ChatGPT, including clear descriptions of their problems and specifying the programming language they are working with. Later, it will be important to not over-specify instructions so that students are allowed to demonstrate their ability to use the tool effectively for learning and problem solving.

Balancing ChatGPT interactions with hands-on coding practice, peer collaboration, and instructor feedback will ensure a comprehensive learning experience. Additionally, educating

© 2025 ITMA. All rights reserved.

students on the importance of using ChatGPT ethically, emphasizing that they should not plagiarize and should always understand and be able to explain the code they use from ChatGPT will be critical.

5. Conclusion

In conclusion, implementing ChatGPT into teaching introductory coding courses can offer a more personalized, engaging, and efficient learning experience. It can create a *de facto* personal digital tutor for the course. This technology can enhance traditional teaching methods, providing students with the tools they need to succeed in the ever-evolving field of programming.

Acknowledgments: ChatGPT 4.0 was used to generate the examples provided herein for each of the five types of exercises. Real ChatGPT prompts and responses are featured in this paper.

References

[1] Sullivan, M.; Kelly, A.; McLaughlan, P. ChatGPT in higher education: Considerations for academic integrity and student learning. *J. Appl. Learn. Teach*. **2023**, 1, 1–10.

[2] Cooper, G. Examining science education in ChatGPT: An exploratory study of generative artificial intelligence. *J. Sci. Educ. Technol.* **2023**, 32.

[3] Crawford, J.; Cowling, M.; Allen, K.A. Leadership is needed for ethical ChatGPT: Character, assessment, and learning using artificial intelligence (AI). *J. Univ. Teach. Learn. Pract.* **2023**.

[4] van Dis, E.A.M.; Bollen, J.; Zuidema, W.; van Rooij, R.; Bockting, C.L. ChatGPT: Five priorities for research. *Nature* **2023**, 614(7947), 224–226.

[5] Strzelecki, A. To use or not to use ChatGPT in higher education? A study of students' acceptance and use of technology. *Interact. Learn. Environ*. **2023**, 1, 1–14.

[6] Ellis, M.E.; Casey, K.M.; Hill, G. ChatGPT and Python programming homework. *Decis. Sci. J. Innov. Educ.* **2024**, 1, 1–14.

© 2025 ITMA. All rights reserved.