EXPLORING STUDENT PERCEPTION AND INTERACTION USING CHATGPT IN PROGRAMMING EDUCATION

Boxuan Ma, Li Chen and Shin'ichi Konomi
Kyushu University, Japan

ABSTRACT

Generative artificial intelligence (AI) tools like ChatGPT are becoming increasingly common in educational settings, especially in programming education. However, the impact of these tools on the learning process, student performance, and best practices for their integration remains underexplored. This study examines student experiences and interactions using ChatGPT in a beginner-level Python programming course through a combination of questionnaire responses and student-ChatGPT dialogue data analysis. The findings reveal a generally positive student reception toward ChatGPT, emphasizing its role in enhancing the programming education experience. Additionally, by clustering and analyzing the types of prompts students use, we identify four distinct patterns of ChatGPT usage and compare the performance outcomes associated with each pattern. This empirical research provides a deeper understanding of AI-enhanced programming education, offering valuable insights and suggesting pathways for future research and practical applications.

KEYWORDS

Generative AI, ChatGPT, Python, Programming Education

1. INTRODUCTION

The advent of generative artificial intelligence (AI) tools, such as ChatGPT, has introduced transformative changes across various sectors, including education (Chen et al., 2024). These tools have the potential to reshape the way educators approach teaching and how students engage with learning materials. In the context of programming education, the integration of AI tools marks a significant shift in how teaching and learning are conceptualized and implemented (Rahman and Watanobe, 2023; Malinka et al., 2023). Traditionally, programming courses have emphasized hands-on practice and the development of problem-solving skills through iterative learning. However, with AI tools now capable of generating code, providing instant feedback, and even debugging, the role of the instructor and the learning process itself is evolving. While these advancements offer new opportunities for personalized learning and efficient problem-solving, they also raise critical questions about the role and effectiveness of AI in enhancing student learning outcomes.

One primary concern voiced by educators and students alike is the potential for over-reliance on AI tools, which could impede the development of critical thinking and problem-solving skills (Tlili et al., 2023). The ease with which AI can generate solutions may discourage students from engaging fully with the problem-solving process, leading to a surface-level understanding rather than a deep mastery of programming concepts (Skjuve et al., 2023). This concern is not unfounded, as the crux of programming education lies in nurturing a mindset capable of breaking down complex problems and devising innovative solutions—skills that might atrophy if students become overly dependent on AI-generated answers. Furthermore, the influence of AI on student behavior and learning processes is still not fully understood, making it essential to investigate these dynamics more thoroughly. This concern underscores the need for a deeper understanding of how AI influences student performance and their overall approach to learning in programming courses. Understanding these questions is crucial for educators to develop strategies that integrate AI tools to support, rather than undermine, the educational objectives of programming courses (Kasneci et al., 2023).

This research investigates the impact of ChatGPT on student experiences and learning outcomes in an introductory Python programming course to address these concerns, it mainly focuses on two research questions:

RQ1: How do students perceive the use of ChatGPT in programming learning, and how does it impact their learning experiences?

RQ2: How do students use ChatGPT, and how do different types of ChatGPT usage influence their performance?

To answer these questions, we first analyze student perceptions and experiences with ChatGPT through questionaries (RQ1). Then, using the student-GPT interaction data, we further analyzed the types of prompts students used to explore how students give prompts to use ChatGPT in class. Through cluster analysis, we identified four different patterns of ChatGPT usage and compared different types of ChatGPT usage's impacts on student performance (RQ2). This study contributes to the growing body of research on AI-enhanced education by offering insights into how ChatGPT affects student learning in programming. Our findings deepen our understanding of AI's role in education and suggest practices for integrating these tools into programming curricula.

2. RELATED WORK

2.1 ChatGPT's Ability to Enhance Programming Education

Recent studies have highlighted ChatGPT's ability to handle common programming challenges with impressive results, such as code generation, program repair, and code summarization (Rajala et al. 2023). Additionally, a recent work (Phung et al. 2023) systematically compared GPT models with human tutors and found that they approach human-level performance in Python programming tasks and real-world buggy programs. In addition, ChatGPT has proven valuable in providing feedback on programming assignments and helping students apply theoretical knowledge in practice. Previous works demonstrated the model's effectiveness in generating personalized feedback, which students rated positively (Pankiewicz and Baker, 2023). Overall, ChatGPT's capabilities as a programming assistant extend beyond automation, showing significant promise in educational applications as well.

2.2 Students' Perception of Using ChatGPT for Programming

Numerous studies have explored students' perspectives on using ChatGPT in programming education (Biswas, 2023; Humble et al., 2023; Yilmaz et al., 2023; Shoufan, 2023; Ma et al., 2024). These studies have recognized both the advantages and drawbacks of employing ChatGPT for programming learning. Their findings indicate that students view ChatGPT as a useful tool, noting its ability to respond quickly and effectively to questions, thereby reducing time spent on researching solutions. Students also appreciate its clear explanations and well-organized responses (Ma et al., 2024), whether they are trying to grasp concepts or solve programming problems, such as identifying errors and guiding corrections (Ghimire and Edwards, 2024). However, the studies also highlight some disadvantages mentioned by students. Many are aware of the occasional incorrectness in ChatGPT's responses or that it may not always provide the answers students need, particularly in programming contexts where multiple solutions exist (Shoufan, 2023; Ma et al., 2024). Additionally, many students are concerned that relying on ChatGPT might make them overly dependent on it, thereby reducing their ability to think critically and learn independently (Yilmaz et al., 2023; Ma et al., 2024).

2.3 Impact of Students Using ChatGPT for Programming

Researchers have also examined whether using ChatGPT influences student performance. Recent studies (Brender et al. 2024) revealed that students who used ChatGPT performed similarly to those who did not have access to the tool. Other research has evaluated the quality of AI-assisted student solutions to programming tasks (Qureshi, 2023). However, the research consistently indicates that while ChatGPT may

enhance task performance and completion rates, it does not significantly deepen understanding (Kazemitabaar et al., 2023). Further investigation is needed to better understand how students interact with ChatGPT and its impact on both learning outcomes and task performance.

3. METHOD

Table 1. Pre-questionnaire and post-questionnaire items

Pre-questionnaire items
Q1 How familiar are you with ChatGPT? • I can explain everything about it • I can explain it to some extent • I somewhat understand it • I don't understand it well • I don't know it at all
Q2 Do you use ChatGPT? • I am using it • I used to use it but not anymore • I haven't used it but plan to use it in the future • I don't use it and have no plans to use it in the future
Q3 What is your level in Python programming? • Beginner • Intermediate • Advanced
Q4 Do you think using ChatGPT to learn Python programming is beneficial or detrimental? • I think it is positive • I rather think it is positive • I am neutral • I rather think it is negative • I think it is negative
Post-questionnaire items
Q1 Is ChatGPT helpful for learning programming? (5-point likert scale)
Q2 Will you keep using ChatGPT for learning programming? (5-point likert scale)
Q3 Will you use ChatGPT often? (5-point likert scale)
Q4 Will you recommend ChatGPT to friends? (5-point likert scale)
Q5 How do you think ChatGPT can help with programming learning? (Multiple answers allowed) • Correct programming code • Answer programming questions • Provide examples of programming code • Offer learning advice and resources • Explain programming concepts
Q7 What are the advantages of using ChatGPT for programming learning? (Open-ended Question)
Q8 What are the limitations or disadvantages of using ChatGPT for programming learning? (Open-ended Question)
Q9 How could ChatGPT be improved to better assist with programming learning? (Open-ended Question)

This study aims to investigate students' experiences with ChatGPT within a university Python programming exercise course and employs a mixed-methods approach with both quantitative and qualitative data to answer our research questions. Students are allowed to use ChatGPT freely throughout the class. They can log into a web-based interface that leverages the GPT-4 API, allowing them to engage in conversations directly through this platform. All the conversations between students and ChatGPT were collected for analysis.

3.1 Participants and Context

A total of 36 undergraduate students enrolled in a Python programming course at our university participated in this study. The participants included 34 freshmen, 1 sophomore, and 1 junior, with 19 identifying as male and 17 as female. IRB approval in our university was obtained for the study. The course, spanning fourteen weeks, covers the fundamentals of the Python programming language. Each week, students attend a 90-minute lesson. The first 45 minutes of each session are dedicated to instruction, utilizing presentations and e-books, followed by 45 minutes of hands-on programming assignments related to that week's topic. Students need to complete programming assignments (e.g., write a specific function based on the requirements) and submit their solutions through the Learning Management System (LMS), which automatically checks and records all interactions and code submissions.

3.2 Data Collection and Analysis

At the beginning of the course, we asked all participants to fill out a pre-questionnaire to collect demographics such as major and gender. We also asked about their programming experiences and their

familiarity with ChatGPT. After the class, we conducted a post-questionnaire, which gathered students' opinions about their experience of using ChatGPT for programming learning. A 5-point Likert scale (1-completely disagree, 5-completely agree) is used to measure different aspects of the students' opinions toward using ChatGPT for learning programming. Also, open questions were developed to determine students' viewpoints on the use of ChatGPT for programming learning purposes. All question items are shown in Table 1. Students were instructed to use ChatGPT freely throughout the class, after each class, they were asked to record their dialogue data with GPT and submit them to LMS. Note that not all participants contributed equally to the dataset, some students did not engage sufficiently with the LMS, resulting in incomplete data for certain weeks. We attempt to answer our research questions by analyzing the data collected from student prompts, ChatGPT responses, and surveys. Through questionnaires, we first explored students' perspectives on the role ChatGPT played. Next, we analyzed ChatGPT prompts obtained from the students and conducted a cluster analysis to find the students' common usage patterns. Finally, we compared the performance of the students in different clusters. The details and results will be described in the next section.

4. RESULTS

4.1 Analysis of Questionnaires

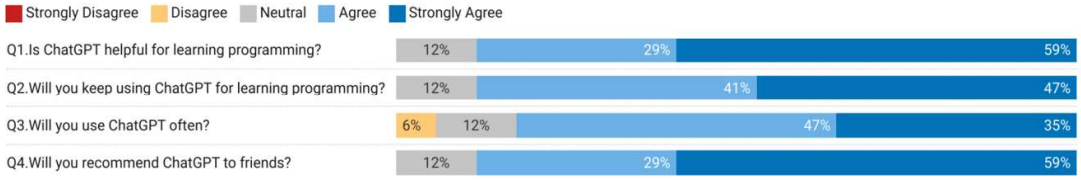

Figure 1. Students' responses to post-questionnaire

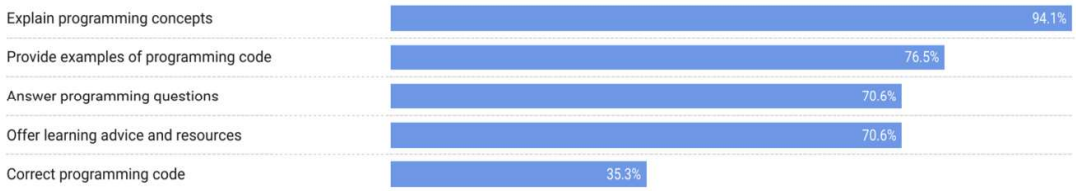

Figure 2. Students' responses on how ChatGPT can help with programming learning

The pre-questionnaire reveals that most respondents are familiar with ChatGPT, a significant portion (86.1%) reported having some knowledge, while 13.9% reported low or no understanding (Q1). This is not surprising as 88.9% of respondents are using ChatGPT or have used it before, and only 11.1% of respondents have not used it yet (Q2). Also, a majority of respondents are beginners in Python programming (75%), and 25% are reported as intermediate or advanced. Overall, the perceived impact of ChatGPT on learning (Q4) was generally positive, with 80.5% viewing it as beneficial to some extent.

Figure 1 shows the post-questionnaire results. There is overwhelming support for ChatGPT as a beneficial tool in learning programming, with most respondents in agreement (Q1, 58.8% strongly agree, 29.4% agree, 11.8% neutral and no respondent disagreed). Similarly, there is a high intention to continue using ChatGPT for learning programming (Q2), with 47.1% strongly agree and 41.2% agree to continue its use. Furthermore, most respondents plan to use ChatGPT frequently in the future for programming learning (Q3, 47.1% strongly agree, and 35.2% agree), and the majority of respondents agree to recommend ChatGPT to their friends (Q4, 58.8% strongly agree and 29.4% agree). This suggests that users find value in ChatGPT's capabilities for their learning needs. Overall, the data reflects a highly positive reception of ChatGPT among learners in programming courses. Additionally, Figure 2 illustrates the perceived usage of

ChatGPT in programming learning. The highest percentage, 94.1%, indicates that ChatGPT helps provide code examples. 76.5% believe that ChatGPT aids in explaining programming concepts. Equal portions of respondents, at 70.6% each, believe that ChatGPT aids in answering programming questions and correcting programming code. This highlights its role in offering practical coding assistance and clarification on tasks. Analysis of the dialogue data shows that students frequently use ChatGPT as a debugging tool, especially when they are working on programming exercise activities. Lastly, 35.3% of respondents value ChatGPT for offering learning advice and resources.

Our open-ended questions revealed that students find ChatGPT useful due to its ability to quickly identify and correct coding errors, provide clear explanations, and offer helpful suggestions. However, they also noted limitations, such as incorrectness in its responses, which is confusing, particularly when its answers differ from those given by teachers. Additionally, students mentioned that ChatGPT sometimes provides overly advanced information, making it difficult to follow. There is also a concern that over-reliance on ChatGPT might hinder the development of fundamental programming skills by encouraging shortcuts. To enhance its effectiveness, students suggested improving the correctness and relevance of ChatGPT's responses, incorporating step-by-step explanations, and aligning its guidance more closely with classroom content.

4.2 Analysis of Interactions

4.2.1 Prompt types

Table 2. Categorized prompt types and descriptions

Prompt Types	Description
Code Verification	Ask for verification the correctness of code.
Conceptual Question	Ask questions about programming-related conceptual knowledge.
Asking for Advice/Learning Resources	Request advice or resources for learning.
Code Correction	Ask for debugging and correction of errors in the code.
Error Message Interpretation	Request interpretation of the meaning and causes of error messages.
Code Implementation Questions	Ask specific questions related to code implementation.
Code Generation	Request the generation of code snippets for specific requirements.
Asking for example	Request example code for specific concepts or functionalities.
Code Optimization	Request the improvement of existing code based on specific needs.
Code Explanation	Ask for explanation of the functionality and purpose of the code.

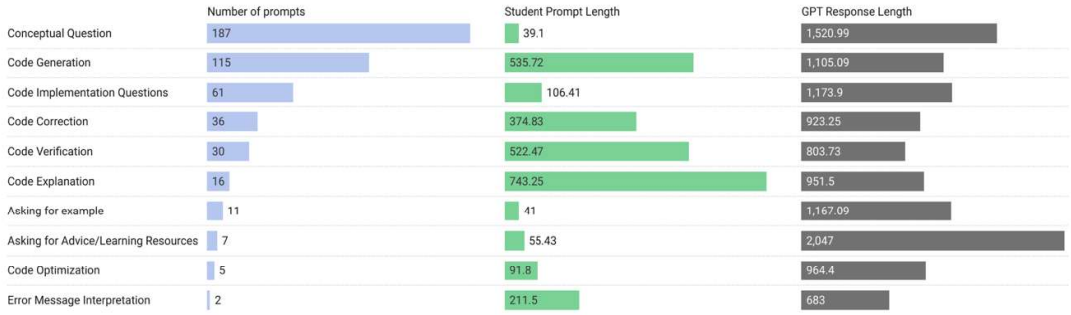

Figure 3. Count of various types and length of prompts and responses

To better understand the utilization of ChatGPT in the class, we analyzed 470 ChatGPT prompts obtained from the students. Although some previous work has categorized student prompts, their classifications were generally broader, often dividing prompts into just three or four categories (Ghimire and Edwards, 2024;

Brender et al. 2024). To gain a more nuanced understanding of student behavior, we conducted a detailed review of the interactions between students and ChatGPT, leading to a more granular classification of student prompts. This resulted in ten distinct categories of prompts, which are summarized in Table 2.

Figure 3 illustrates the distribution of various types of prompts submitted by students. Conceptual questions, requests for code generation, and code implementation questions were the most common types of prompts. The figure also shows the average length of prompts sent by students. We found that the average prompt lengths for code explanation, code generation, code verification, and code correction are significantly higher than those for other prompt types. This is expected, as students typically include the code itself in these prompts. In contrast, other types of prompts tend to have an average length of fewer than 200 characters. In contrast to the diversity in the average length of prompts, we found that all prompts received fairly long responses (more than 600 characters). A careful examination reveals that GPT provides detailed answers across different prompt types, often including examples and explanations, underscoring its role as a comprehensive and supportive tool in the learning process. Among these, prompts asking for advice or resources elicited the longest GPT responses (2,047 characters), while conceptual questions received the second longest responses, averaging 1,520.99 characters.

4.2.2 Cluster Analysis

Table 3. Clustering results of students' prompts

Prompt Types	Conceptual Learners (C1)	Code Verifiers (C2)	Practical Coders (C3)	AI-Reliant Coders (C4)
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Code Verification	2.02% (0.21)	**80.00% (1.41)**	4.33% (0.35)	6.48% (0.54)
Conceptual Question	**75.60% (1.44)**	0.00% (0.00)	7.39% (0.39)	1.03% (0.14)
Asking for Advice/Learning Resources	1.88% (0.14)	0.00% (0.00)	**8.75% (0.36)**	2.05% (0.29)
Code Correction	3.70% (0.20)	10.00% (0.71)	**21.03% (0.41)**	10.19% (0.69)
Error Message Interpretation	0.31% (0.04)	0.00% (0.00)	**3.17% (0.34)**	0.00% (0.00)
Code Implementation Questions	7.64% (1.21)	0.00% (0.00)	**33.86% (0.98)**	14.71% (0.31)
Code Generation	2.18% (0.15)	10.00% (0.71)	11.63% (0.18)	**62.24% (1.67)**
Asking for example	**5.01% (0.19)**	0.00% (0.00)	0.00% (0.00)	0.00% (0.00)
Code Optimization	0.33% (0.03)	0.00% (0.00)	**1.43% (0.12)**	0.00% (0.00)
Code Explanation	1.33% (0.09)	0.00% (0.00)	**8.41% (0.20)**	3.30% (0.17)

Table 4. The percentage of students in each cluster

Category	C1	C2	C3	C4
Percentage of students	46.2%	11.5%	23.1%	19.2%

Following previous work (Brender et al. 2024), we calculated for each student the percentage of prompts in each category and then used k-means clustering. In order to identify the optimal number of clusters, we have performed the elbow method, and the number of clusters is set to 4 based on the results. Table 3 shows the results including four clusters: Conceptual Learners (C1), Code Verifiers (C2), Practical Coders (C3), and AI-Reliant Coders (C4), and the percentage of students in each cluster are shown in Table 4. We give each cluster a name based on the results.

Conceptual Learners: Cluster 1 is dominated by students who primarily focus on understanding programming concepts. These students are less concerned with debugging or validating their code, likely preferring to focus on the broader conceptual frameworks of programming. With 75.60% of their prompts categorized under Conceptual Questions, they are heavily invested in deepening theoretical understanding.

Code Verifiers: Cluster 2 is characterized by a strong emphasis on ensuring code correctness, with a striking 80.00% of prompts focused on Code Verification. These students are highly detail-oriented, often seeking to validate their code and ensure it functions as intended. This cluster also shows a moderate engagement with Code Correction (10.00%) and Code Generation (10.00%), while the complete lack of

engagement in Conceptual Questions and other categories suggests that these students prioritize practical coding tasks over theoretical learning.

Practical Coders: Cluster 3 includes students who exhibit a balanced approach, with significant engagement across several categories, particularly Code Implementation Questions (33.86%) and Code Correction (21.03%). These students are highly practical, focusing on the application of their coding knowledge and the refinement of their code. They also show notable involvement in Code Explanation (8.41%), indicating that they often seek to understand the purpose and functionality of their existing code.

AI-Reliant Coders: Cluster 4 is named AI-Reliant Coders, with 62.24% of their prompts centered on code generation. After examining their dialogue details, we found that these students frequently copy and paste their coding problems directly into ChatGPT, depending on the AI to generate entire solutions with minimal manual input. When issues arise, they often rely on GPT to correct errors and validate the code, until their code successfully passes, this finding is aligned with previous work (Qureshi, 2023). This is also evident in their moderate engagement with Code Implementation Questions (14.71%) and Code Correction (10.19%), showcasing their preference for using GPT for generating code and also using GPT for refining and debugging it.

4.2.3 Student Academic Performance

Table 5. Students' performance of different clusters

	Category	C1	C2	C3	C4
Assignments (Full marks=150)	Mean (SD)	144.85 (7.94)	147.48 (3.01)	144.80 (5.66)	148.27 (0.97)
Final Grade (Full marks=5)	Mean (SD)	4.42 (0.51)	4.00 (0.00)	4.50 (0.55)	4.00 (0.00)

Table 5 shows students' performance across different clusters. It includes data on the total assignment scores (out of 150) accumulated over 12 weeks and final grades (with letter grades S, A, B, C, F converted to numerical values 5, 4, 3, 2, 1), with both mean and standard deviation provided for each cluster. Given the non-normal distribution of the data, we used the Mann-Whitney U test to compare the performance between every two groups. For assignments, AI-Reliant Coders (C4) and Code Verifiers (C2) outperformed those in the other clusters, achieving higher mean scores. For AI-Reliant Coders, who likely relied heavily on ChatGPT for generating solutions directly translates to high performance in assignments, as the AI's output is often correct. Code Verifiers appear to have written their code and then used ChatGPT for verification and debugging. This allows them to correct errors before submission, which significantly increases the accuracy of their code. However, the final course grades, which provide a more holistic evaluation of student performance, revealed that Practical Coders (C3) and Conceptual Learners (C1) earned higher grades compared to C2 and C4. They achieved higher final grades likely due to their comprehensive understanding of concepts and balanced engagement in both theoretical and practical aspects of the course, which are key components of a holistic evaluation beyond just assignment scores. While the Mann-Whitney U test indicated no statistically significant differences in performance across the clusters, these findings offer valuable insights into integrating AI tools in programming courses.

5. CONCLUSION

This study delves into ChatGPT's role in Python programming learning, focusing on student perceptions and interactions. Through questionnaires, we first explored students' perspectives on the role ChatGPT played. Next, we analyzed ChatGPT prompts obtained from the students and conducted a cluster analysis to find the students' common usage patterns. Finally, we compared the performance of the students in different clusters. By analyzing the questionnaire data, our research provides insights into effectively integrating LLMs into programming education. The results suggest that students value ChatGPT as a beneficial tool that assists them and enhances their learning experiences in program learning. Interaction data analysis indicated detailed interaction between students and ChatGPT.

There are some limitations to this study. First, the sample size is relatively small, which limits the generalizability of the results. Second, while our analysis methods identified different patterns, these clusters

may not accurately represent students' true intent, as they might be influenced by factors such as the ability to provide high-quality prompts, the context in which the prompts were used, and student's prior knowledge and experience. Further validation with additional data is required to confirm these findings. In addition, although we compared the performance of different students, the absence of a closed-format test limits the results. For future work, we plan to collect and analyze more data to further explore the effects on student behavior and learning outcomes. We will also focus on a detailed examination of students' code quality after using ChatGPT, as well as an in-depth analysis of ChatGPT's responses.

ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Numbers JP20H00622 and JP24K20903.

REFERENCES

Biswas, S. (2023). *Role of ChatGPT in computer programming.: ChatGPT in computer programming.* Mesopotamian J. Comput. Sci. 2023, 8–16.

Brender, J., El-Hamamsy, L., Mondada, F., & Bumbacher, E. (2024). *Who's Helping Who? When Students Use ChatGPT to Engage in Practice Lab Sessions.* In *International Conference on Artificial Intelligence in Education* (pp. 235-249). Cham: Springer Nature Switzerland.

Chen, L., Li, G., Ma, B., Tang, C., Okubo, F., Shimada, A. (2024). *How do strategies for using ChatGPT affect knowledge comprehension?.* Springer, Cham. In *International Conference on Artificial Intelligence in Education* (pp. 151-162). Cham: Springer Nature Switzerland.

Ghimire, A., & Edwards, J. (2024). *Coding with AI: How are tools like chatgpt being used by students in foundational programming courses.* In *International Conference on Artificial Intelligence in Education* (pp. 259-267). Cham: Springer Nature Switzerland.

Humble, N., et al. (2023). *Cheaters or AI-enhanced learners: Consequences of ChatGPT for programming education.* Electron. J. e-Learn.

Kasneci, E., et al. (2023). *ChatGPT for good? on opportunities and challenges of large language models for education.* Learn. Individ. Differ. 103, 102274.

Kazemitabaar, M., Chow, J., Ma, C.K.T., Ericson, B.J., Weintrop, D., Grossman, T. (2023). *Studying the effect of AI code generators on supporting novice learners in introductory programming*. In: CHI'2023, pp. 1–23. ACM.

Ma, B., Chen, L., Konomi, S. (2024). *Enhancing programming education with ChatGPT: a case study on student perceptions and interactions in a Python course.* In *International Conference on Artificial Intelligence in Education* (pp. 113-126). Cham: Springer Nature Switzerland.

Malinka, K., Peres´ını, M., Firc, A., Hujn´ak, O., Janus, F. (2023). *On the educational impact of ChatGPT: is artificial intelligence ready to obtain a university degree?* In: Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education, vol. 1. pp. 47–53.

Pankiewicz, M., Baker, R.S. (2023). *Large language models (GPT) for automating feedback on programming assignments.* arXiv preprint arXiv:2307.00150.

Phung, T., et al. (2023). *Generative AI for programming education: benchmarking ChatGPT, GPT-4, and human tutors*. Int. J. Manag. 21(2), 100790.

Qureshi, B. (2023). *Exploring the use of ChatGPT as a tool for learning and assessment in undergraduate computer science curriculum: opportunities and challenges.* In: 9th International Conference on e-Society, e-Learning and e-Technologies, pp. 7–13. https://doi.org/10.1145/3613944.3613946

Rahman, M.M., Watanobe, Y. (2023). *Chatgpt for education and research: opportunities, threats, and strategies.* Appl. Sci. 13(9), 5783.

Rajala, J., Hukkanen, J., Hartikainen, M., Niemel¨a, P. (2023). *"Call me Kiran"-ChatGPT as a tutoring chatbot in a computer science course.* In: Proceedings of the 26th International Academic Mindtrek Conference, pp. 83–94.

Shoufan, A. (2023). *Exploring students' perceptions of ChatGPT: thematic analysis and follow-up survey*. IEEE Access.

Skjuve, M., Følstad, A., Brandtzaeg, P.B. (2023). *The user experience of ChatGPT: findings from a questionnaire study of early users.* In: Proceedings of the 5th International Conference on Conversational User Interfaces, pp. 1–10.

Tlili, A., et al. (2023). *What if the devil is my guardian angel: Chatgpt as a case study of using chatbots in education.* Smart Learn. Environ. 10(1), 15.

Yilmaz, R., Yilmaz, F.G.K. (2023). *Augmented intelligence in programming learning: examining student views on the use of ChatGPT for programming learning.* Comput.Hum. Behav. Artif. Hum. 1(2), 100005.