International Journal of Educational Technology in Higher Education

RESEARCH ARTICLE Open Access

Would ChatGPT-facilitated programming mode impact college students' programming behaviors, performances, and perceptions? An empirical study

Dan Sun[1], Azzeddine Boudouaia[2], Chengcong Zhu[3] and Yan Li[2*]

*Correspondence:
yanli@zju.edu.cn

[1] Chinese Education Modernization Research Institute, Hangzhou Normal University, Yu Hang Tang Rd 2318, 311121 Hangzhou, China
[2] Zhejiang University, College of Education, Yu Hang Tang Rd 866, 310058 Hangzhou, Zhejiang, China
[3] Xiaoshan High School, Gongxiu Rd 538, 311201 Hangzhou, China

Abstract

ChatGPT, an AI-based chatbot with automatic code generation abilities, has shown its promise in improving the quality of programming education by providing learners with opportunities to better understand the principles of programming. However, limited empirical studies have explored the impact of ChatGPT on learners' programming processes. This study employed a quasi-experimental design to explore the possible impact of ChatGPT-facilitated programming mode on college students' programming behaviors, performances, and perceptions. 82 college students were randomly divided into two classes. One class employed ChatGPT-facilitated programming (CFP) practice and the other class utilized self-directed programming (SDP) mode. Mixed methods were utilized to collect multidimensional data. Data analysis uncovered some intriguing results. Firstly, students in the CFP mode had more frequent behaviors of debugging and receiving error messages, as well as pasting console messages on the website and reading feedback. At the same time, students in the CFP mode had more frequent behaviors of copying and pasting codes from ChatGPT and debugging, as well as pasting codes to ChatGPT and reading feedback from ChatGPT. Secondly, CFP practice would improve college students' programming performance, while the results indicated that there was no statistically significant difference between the students in CFP mode and the SDP mode. Thirdly, student interviews revealed three highly concerned themes from students' user experience about ChatGPT: the services offered by ChatGPT, the stages of ChatGPT usage, and experience with ChatGPT. Finally, college students' perceptions toward ChatGPT significantly changed after CFP practice, including its perceived usefulness, perceived ease of use, and intention to use. Based on these findings, the study proposes implications for future instructional design and the development of AI-powered tools like ChatGPT.

Keywords: ChatGPT, Programming learning, Behavioral analysis, Perception, College student

Introduction

Programming education has become increasingly important in the current higher education system because it could promote college students' computational thinking skills, which are vital in various working situations (Jancheski, 2017; Stehle & Peters-Burton, 2019). However, the endeavor to harness the advantages of programming is not devoid of obstacles. Extensive research has determined that students actively engaged in programming education may encounter a range of challenges (Looi et al., 2018; Sun et al., 2021a). The challenges that were faced included a deficiency in pertinent technical skills and obstacles hindering the ability to access crucial resources (Bau et al., 2017; Tom, 2015). The challenges that college students face in their programming learning can have a negative impact on their overall educational experience. In this regard, it seems necessary to offer comprehensive support to college students in their programming learning process (Lu et al., 2017; Sun et al., 2021d).

The use of certain technological enablers, for instance, interactive coding platforms, integrated development environments (IDEs), and online coding communities, has a substantial impact on the level of programming performance and behavior. They may lead to promoting the acquisition of programming skills, facilitating their practical application, and catering to the personalized preferences of programming learners (Chevalier et al., 2020; Ghatrifi et al., 2023; Nurbekova et al., 2020). An exemplary illustration of programming education can be found in the utilization of ChatGPT. The impact of ChatGPT on learning has been remarkable, as it has brought about a revolutionary wave of technological advancements that have greatly facilitated the teaching and learning processes (Firaina & Sulisworo, 2023; Lo, 2023). ChatGPT may be a resource for determining performance, learning, and discussing numerous programming-related topics. ChatGPT can formulate, clarify, and illustrate code samples in response to students' inquiries. It has the potential to serve as an exhibition platform for various programming solutions, explanations of various methodologies, and illustrative approaches (Chen et al., 2023; Yilmaz & Yilmaz, 2023a). ChatGPT can generate a customized learning trajectory based on the learner's existing proficiency and desired objectives in programming learning. ChatGPT can provide recommendations for online courses, tutorials, books, and many other resources to enable ongoing personal and professional development (Jalil et al., 2023; Surameery & Shakor, 2023; Tian et al., 2023). ChatGPT exhibits exceptional characteristics, including the capability to produce text that faithfully replicates authentic human dialogue when given inputs. This feature is very different from traditional teacher-led instruction, and it makes people think about how it could be used and incorporated into programming education (Firaina & Sulisworo, 2023; Javaid et al., 2023). Further significant factors exist that necessitate scrutiny. Notably, conventional teaching methods do not significantly impact the accessibility of vital information; rather, it is ChatGPT's sophisticated algorithmic architecture and computational prowess that ensure its availability. In addition, it's important to acknowledge that there may be some inaccuracies in the error detection system of ChatGPT, which could potentially impact students' motivation and capacity to utilize the feedback provided.

What learning mode is more effective and preferred by college students for programming learning remains uncertain. This uncertainty is partly attributed to programming learning studies to date have been based on the comparison between the most commonly used technological sources and devices and traditional learning forms. As an illustrative instance, researchers have juxtaposed and contrasted various programming methodologies. Sun et al. (2021a), for instance, attempted to harmonize text-based programming techniques with isomorphic block-based techniques. With instructor-led delivery, Sun et al. (2021b) sought to reconcile learner-oriented, unplugged programming. Sun et al. (2021d) facilitated the lectures on novice programming instruments and the transition from Logo to Scratch. To address these obstacles, teachers have implemented innovative pedagogical approaches, including game-based learning, offline instruction, and project-centric programming, which are commonly used in informal learning environments. The primary aim of these methodologies is to convert traditional instructor-led programming instruction into interactive, learner-focused programming activities (Brackmann et al., 2017; Hosseini et al., 2019; Nurbekova et al., 2020).

For this study, we utilized Davis' (1989) technology acceptance model (TAM) as a theoretical framework to investigate students' acceptance of ChatGPT in programming education. Researchers have used Davis' (1989) technology acceptance model as a framework to look into how students interact with AI-based learning technologies (Xie et al., 2023) by looking at the currently published academic literature. Researchers extensively employ the TAM framework in studies focused on the domain of programming learning. Studies conducted by Cheng (2019) and Thongkoo et al. (2020) have provided evidence that supports the claim that the TAM is a valid predictive model for technological adoption in programming learning. In today's modern era, the widespread use of computing technology has made it an essential tool for students. It has become so prevalent that it is now considered indispensable, enabling students to take charge of their learning and strive toward long-term objectives such as skill enhancement (Peng et al., 2023). Developed by Davis (1989), TAM aims to shed light on the factors that influence user behavior in adopting new technologies (Xie et al., 2023). According to the TAM, users' perceptions of a technology's usefulness and ease of use play a vital role in determining their adoption and regular usage of it (Davis, 1989). In addition, the attitudes and behavioral intentions of users play a crucial role in determining their willingness to adopt technology in the learning process (Yang & Tsai, 2008; Yi et al., 2016). Perceived ease of use is a concept that revolves around how users assess the simplicity or complexity of a technological device or system based on their impressions and expectations (Davis, 1989). When considering the value of a particular technology, it is important to take into account an individual's assessment of how it will enhance their work efficiency (Xie et al., 2023). When a new piece of technology is perceived as user-friendly, the chances of people embracing it are significantly higher. Nevertheless, individuals tend to become less inclined to utilize technology when they encounter challenges in acquiring the necessary skills to operate it (Teo et al., 2008).

It will be worth exploring the effectiveness of ChatGPT in programming learning. In this regard, this study aimed to explore the impact of ChatGPT-facilitated programming on college students' programming behaviors, performance, and perception by comparing two kinds of learning modes: ChatGPT-facilitated programming (CFP) mode and traditional self-directed programming (SDP) mode. There were four research questions:

RQ1: What are the differences in the programming behaviors of students engaged in CFP mode compared with those in SDP mode?

RQ2: What are the differences in the programming performances of students engaged in CFP compared with those in SDP mode?

RQ3: How do college students describe their user experiences with ChatGPT in CFP mode?

RQ4: How do college students' perceptions of ChatGPT change following their experience with CFP mode?

The results of this study offer significant insights that can inform policy-making regarding the most effective approaches to expand and support programming education at the college and university levels. Despite being conducted within the specific context of China, our research findings and consequences hold significance for scholars, policymakers, and practitioners worldwide. Countries, regardless of their level of development, can recognize and rectify possible deficiencies in programming education through careful consideration of the data presented in our research. The advent of AI tools such as ChatGPT introduces novel complexities to programming instruction. Our research makes a significant and innovative contribution to this urgent matter at an international level. The ramifications transcend academic settings and universities, encompassing the approaches and policies of governments across the globe. In conclusion, the findings of this inquiry substantially advance our comprehension of the perspectives held by aspiring teachers worldwide regarding the application of ChatGPT in computer programming instruction.

Literature review

Advancing programming education with AI-technologies

Technology-enhanced programming learning has been gaining momentum in recent years with the advance of AI technologies. This literature review examines the new trends in the programming and learning fields that have emerged in the age of AI. One of the significant trends in programming education is the integration of AI-powered tools such as chatbots, intelligent tutoring systems, and automated programming assessment software. These tools offer students personalized instruction and immediate feedback to help them progress at their own pace and improve their programming skills. For example, Skalka et al. (2021) proposed a conceptual framework that combines micro-learning and automatic evaluation of source code to give students immediate feedback and involve them in software development in a virtual learning environment. This framework was shown to significantly improve the results of students in advanced programming courses. In the Programming 1 course, Malik et al. (2022) introduced a chatbot that was specifically engineered to highlight problem-solving strategies, common programming errors, syntax, and semantics, with the ultimate goal of assisting inexperienced learners in simultaneously mastering a variety of competencies. The students perceived the chatbot's methodology as advantageous in the above-mentioned points. Klasnja-Milievi et al. (2016) also designed an Intelligent Tutoring System called Programming-Tutor that uses AI to provide an immersive learning experience for online programming courses in the Pacific. This ITS is expected to help students learn programming more easily and efficiently in an online mode and provide valuable formative assessment while enhancing student learning.

Furthermore, the application of data analytics, machine learning, and natural language processing technologies in programming education has gained popularity. These innovations can help analyze programming-related data, extract valuable insights into learners' programming skills, and pinpoint areas where improvement is needed to enable effective feedback and instruction. For instance, Khan et al. (2019) developed a model to predict the performance of introductory programming students based on their early semester grades. The researchers used WEKA to compare eleven different machine-learning approaches and found that the Decision Tree algorithm did the best in terms of recognizing instances, being accurate on the F-measure, and finding true positives. This model is expected to enable students to forecast their probable final grades, empowering them to modify their study approaches for better academic results. In a similar vein, Sivasakthi et al. (2017) developed a predictive data mining model that utilized classification algorithms to predict the performance of first-year Computer Application bachelor's degree students in introductory programming. The research ascertained the most efficient classification algorithm and demonstrated the accuracy of each algorithm in use. In addition, Shen et al. (2023) proposed a student profile model that includes code information and other student characteristics as input to a deep neural network for performance prediction and found that a four-layer deep neural network using all available dimensions of student profiles achieved the best performance.

Despite these promising trends, challenges remain in effectively integrating AI technologies into the learning process. These challenges include ethical considerations such as privacy, security, and bias, and tensions between the use of AI-powered tools and the human aspect of programming involving creativity and problem-solving skills (Gervasi et al., 2021; Mousavinasab et al., 2021; Wang et al., 2023). Recent advancements in large-scale language modeling and AI, such as interactive text generators capable of responding to user prompts (Yilmaz & Yilmaz, 2023c), have not been adequately explored or recognized in programming education. This leaves a gap in implementing state-of-the-art technologies in programming education pedagogy.

Integrating ChatGPT in the programming filed

Its unique features distinguish ChatGPT from conventional programming utilities. Surameery and Shakor (2023) and Yilmaz and Yilmaz (2023c) emphasize that ChatGPT, as an AI language model, enhances its usability for individuals without prior programming knowledge by engaging in conversational language with users. Ray et al. (2014) commonly portray conventional programming resources, such as programming environments, programming languages, libraries, and associated components, as requiring programming expertise and often focusing on a single language. A Python integrated development environment purposefully developed for Python requires users to possess a foundational understanding of this programming language. In contrast, multi-platform integrated development environments such as Flutter or Microsoft Visual Studio support multiple programming languages. In light of this, ChatGPT employs machine learning and natural language processing technologies to accurately interpret and respond to user inputs expressed in colloquial language. In contrast to conventional programming languages, which adhere to particular syntax and principles, ChatGPT is designed to comprehend user inquiries and provide appropriate responses, as explained by OpenAI (2023). According to Jalil et al. (2023), ChatGPT presents a novel programming pedagogical approach that is suitable for a diverse range of individuals, including teachers and students.

However, there are conflicting views on the effectiveness of ChatGPT in programming education. On one hand, Yilmaz and Yilmaz (2023b) highlight the various advantages that ChatGPT offers over other programming tools, such as utilizing natural language, providing easy access, offering quick response times, facilitating personalized learning, supporting multiple languages, offering clear explanations and programming examples, allowing for inquiries and searches, and providing resources for advanced topics. According to Yilmaz and Yilmaz (2023a), ChatGPT distinguishes itself from other programming tools through its utilization of colloquial language, user-friendly interfaces, support for multiple languages, comprehensive search capabilities, provision of lucid explanations accompanied by practical illustrations, facilitation of personalized learning experiences, and engagement with complex subject matters. Yilmaz and Yilmaz (2023c) conducted a study that found that the utilization of generative AI tools, including ChatGPT, could potentially enhance students' morale, programming confidence, and aptitude for computational reasoning. Additionally, Chen et al. (2023) argue that ChatGPT's integration into programming instruction can significantly improve students' programming skills through the provision of code explanations and problem-solving assistance. Jalil et al. (2023) demonstrate that learners' efficacy improves when implementing ChatGPT in software testing instruction. Scientific literature frequently addresses the potential of ChatGPT to assist programmers with support and problem resolution (Surameery & Shakor, 2023; Tian et al., 2023). A multiplicity of scholars, including Lo (2023), Qureshi (2023), and Tlili et al. (2023), have underscored the manifold advantages of ChatGPT. The benefits encompass constant accessibility from any location, promptness, and precision in response, and an interface that is easy for users to navigate.

Although ChatGPT offers numerous advantages in the realm of programming education, it is not without its limitations. The limitations emphasized by Yilmaz and Yilmaz (2023a) include disorganized learning, excessive reliance on external tools and environments as a result of the lack of integrated applications, limited support for data structures and algorithms, and inadequate graphical user interface assistance. In their study, Rahman and Watanobe (2023) highlight the following drawbacks of ChatGPT in programming education: a lack of capability in reasoning through simple logic, potential biases, difficulties in complex logical reasoning, and an inability to process visual information. Moreover, they emphasize the intricate and multifaceted nature of ethical considerations associated with ChatGPT, including but not limited to bias, discrimination, privacy, security, technology misuse, accountability, transparency, and societal repercussions.

Given the numerous variables that may influence the efficacy of ChatGPT in programming instruction, it is imperative to conduct a comprehensive, multifaceted investigation into ChatGPT's function in programming. This will yield fruitful research outcomes, analytical ramifications, and actionable suggestions. However, researchers have conducted the majority of the extant research on the use of ChatGPT in programming education outside the formal programming curriculum. The lack of a controlled comparison or quasi-experimental design in previous studies complicates researchers' ability to make definitive claims regarding the effectiveness of the method in assisting individuals with coding learning. Additionally, there has been minimal involvement by students in both the practice and reflection of ChatGPT in programming learning. As a result, there is a gap in knowledge and practical application of the impact of ChatGPT on programming education.

Research methodology

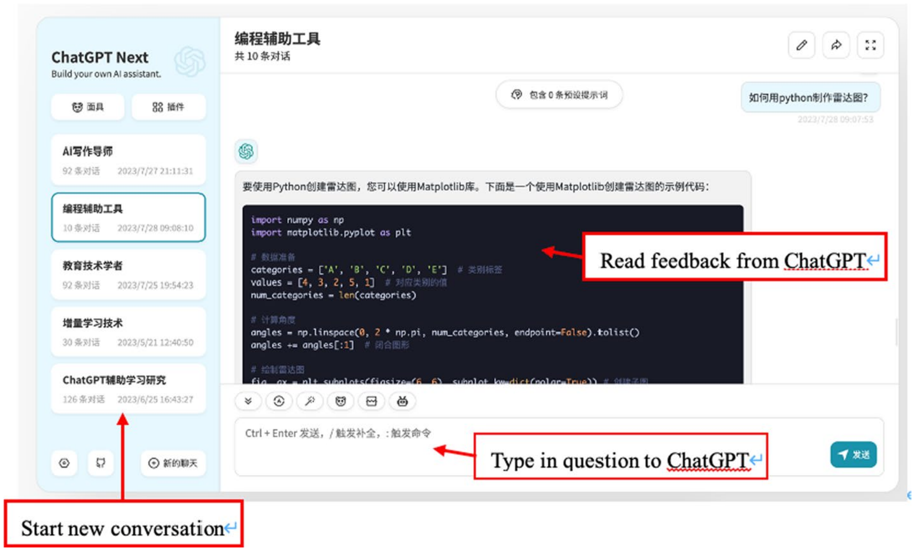
Research context, participants, and instructional procedures

As an integral component of a mandatory "Python Programming" course, we conducted the research specifically tailored for Educational Technology majors enrolled at a Chinese university during the spring term of 2020. We employed a quasi-experimental design to compare the programming behaviors, performances, and perspectives of learners between the control condition (self-directed programming, SDP) and the experimental condition (ChatGPT-assisted programming, CFP) mode. The CFP experimental class consisted of 43 individuals (19 females and 13 males), as opposed to the SDP control class of 39 individuals (16 females and 15 males). With prior knowledge of C programming, the first author of this study, who also instructed both groups, shared this expertise. After obtaining written consent from the review committee, the researchers collected and interacted with data, ensuring no ethical violations occurred. The same instructor oversaw both classes and employed identical instructional strategies, course materials, and guidelines. The sole distinction lay in the integration of ChatGPT into the experimental setup. The instructor, with the assistance and direction of the research team, organized the course into three phases and five eighty-minute learning sessions.

There were 39 learners (female = 16; male = 15) in the control SDP class and 43 learners (female = 19; male = 13) in the experimental CFP class. The students with C programming experience were instructed by the same teacher (the first author of this paper), and both classes were not informed of the different treatments. The review committee's written agreement was obtained to interact with and collect data for research purposes without ethical issues. Classes were taught by the same instructor (the fourth author), who maintained the same teaching style under two conditions, offered the same instructional materials to learners, and used the same teaching guidance for each class, except for the use of ChatGPT in the experiment condition. With guidance and support from the research team, the instructor divided the course into three phases and five instructional sessions (each session lasted 80 min). In Phase I and Phase II, the instructor taught the first four sessions' contents, including the basic concepts of Python programming, including the introduction of Python (e.g., IDLE, input(), eval(), print()), data structure (e.g., int, float, set, list, dictionary), control structure (e.g., if, for, while), functions, and methods (e.g., Recursion, Lambda). In Phase III, the instructor taught the last session's content, which was about a comprehensive programming project (radar chart). The design of the instructional sessions referred to the book titled Python Programming (ICOURSE, 2023). During Phase I and Phase II, both classes received instructor' lecturing with oral presentations and Python demonstrations in first place, followed by self-directed practices with the programming tasks demonstrated before. In Phase III, students were required to complete a radar chart task within 100 min. Students in the SDP mode solved problems according to their knowledge (e.g., refer to additional materials). While in the CFP mode, the ChatGPT Next platform, which was deployed by the first author, was adopted as a major tool to facilitate students' problem-solving process (see Fig. 1). On this platform, gpt-3.5-turbo was chosen as the model, and students can initiate various thematic conversations with ChatGPT by typing their questions in the main window and getting feedback from ChatGPT.

Data collection and analysis approaches

Four methods were used to collect and analyze the data for this study. We documented the programming behaviors of the students by monitoring platform logs and recording computer screen videos (excluding audio). For research facilitation, we selected video recordings of the data obtained during the final class session (specifically, the radar chart task), which averaged 40 min per learner, resulting in a total of 3280 min. We selected the final session for data collection for two primary reasons: first, it focused on a comprehensive programming task that provided a holistic view of how ChatGPT integrated into resolving programming challenges and demonstrating programming performance; and second, informal observation revealed that students in the experimental group had developed performance in utilizing ChatGPT and were more actively involved in this session. By utilizing clickstream analysis (Filva et al., 2019) on this data, we were able to discern the programming behaviors of the students. We conducted the video analysis using a recurrent coding procedure that adhered to a previously approved coding framework (Sun et al., 2021a, b, c, d). Following an initial individual evaluation of the screen-captured videos by two coders who were well-versed in video analysis, they identified preliminary codes representing programming behaviors. Subsequently, the coders collaboratively discussed these codes to arrive at a final coding arrangement, as detailed in Table 1.


ChatGPT Next
Build your own AI assistant.
面具
插件
AI写作导师
92 条对话 2023/7/27 21:11:31
编程辅助工具
10 条对话 2023/7/28 09:08:10
教育技术学者
92 条对话 2023/7/25 19:54:23
增量学习技术
30 条对话 2023/5/21 12:40:50
ChatGPT辅助学习研究
126 条对话 2023/6/25 16:43:27
新的聊天
编程辅助工具
共 10 条对话
包含 0 条预设提示词
2023/7/28 09:07:53
Ctrl + Enter 发送，/ 触发补全，: 触发命令
发送
Read feedback from ChatGPT↵
Type in question to ChatGPT↵
Start new conversation↵

如何用python制作雷达图?

要使用Python创建雷达图，您可以使用Matplotlib库。下面是一个使用Matplotlib创建雷达图的示例代码：

```
import numpy as np
import matplotlib.pyplot as plt

# 数据准备
categories = ['A', 'B', 'C', 'D', 'E']  # 类别标签
values = [4, 3, 2, 5, 1]  # 对应类别的值
num_categories = len(categories)

# 计算角度
angles = np.linspace(0, 2 * np.pi, num_categories, endpoint=False).tolist()
angles += angles[:1]  # 闭合图形

# 绘制雷达图
fig, ax = plt.subplots(figsize=(6, 6), subplot_kw=dict(polar=True))  # 创建子图
```

Fig. 1 ChatGPT-facilitated programming platform used in CFP mode

Table 1 Coding framework for programming behaviors

Code	Description
Understanding task (UT)	A student transferred to the task window to understand the programming projects
Coding in Python (CP)	A student wrote codes with the Python language in the system
Debugging in Python (DP)	A student debugged in PyCharm
Understanding Python codes (UPC)	A student attempted to understand the code with the mouse moving back and forth on the code
Checking Radar Chart (CRC)	A student checked the radar chart output in PyCharm
Reading console message (RCM)	A student read error messages in output console in PyCharm
Asking new questions (ANQ)	A student asked new questions in ChatGPT/browser
Pasting console message (PCM)	A student pasted error messages from output console window in ChatGPT/browser
Pasting Python codes (PPC)	A student pasted Python codes to ChatGPT/browser
Reading feedback (RF)	A student read feedback in ChatGPT/browser
Copy and paste codes (CPC)	A student copy and paste codes from ChatGPT/browser
Referring to additional materials (RAM)	A student referred to additional materials from instructors
Failure in ChatGPT (FC)	ChatGPT failed to give feedback due to technical problems (e.g., cannot connect to the server, get stuck)
Idle operation (IO)	A student had no operation

Based on this coding framework, each coder autonomously recorded the data in chronological order, annotating learner behaviors every 10 s and validating the results with one another. Additionally, we employed lag-sequential analysis (LsA) to assess the behavioral patterns of the learners, using the video coding outcomes as a foundation (Faraone & Dorfman, 1987). Evaluating the frequency of transitions between two behaviors and network representations displayed in two instructional modes was entailed. We chose Yule's Q to represent the extent of transitional associations because it has descriptive value and can account for base numbers of contributions (ranging from -1 to $+1$, with zero indicating no association). This study used a network visualization technique previously developed (Chen et al., 2017) to illustrate the outcomes of LsA in networks. Nodes represent behavior codes along with their corresponding frequencies, while edges denote transitional Yule's Q values. An arrow pointing in the opposite direction of the node denotes the direction of the transition.

Secondly, we evaluated the students' programming performance by assessing their programming assignments. The programming task involved developing a radar chart, with an advanced requirement being the creation of a Holland radar chart and a simplified radar chart serving as the primary requirements. The first author assessed the scores on a scale of one hundred using a rubric that included aspects such as code correctness, programming project aesthetics, and functional integrity. A T-test was employed to compare the academic performance of students enrolled in the two courses.

Thirdly, we conducted semi-structured post-class interviews with students, specifically focusing on their experiences using ChatGPT to practice programming (the interview instrument can be found in Appendix A). We followed a rigorous procedure of thematic analysis to examine the interview data (Cohen et al., 2013). The methodology consisted of the subsequent steps: (1) getting interview transcripts ready for analysis; (2) having two separate coders assign codes to different parts of the data; (3) recording the coded segments so that they can be analyzed later; (4) talking about and comparing parts of the data that have the same codes; (5) aligning and improving the codes to make themes that make sense; and (6) checking the validity and reliability of the themes that were made. Transcripts containing statements such as "I appreciated the human-like explanations ChatGPT provided whenever I made a syntax error; it facilitated my learning pace" and coded segments including "ChatGPT provided human-like explanations; it facilitated my learning pace; identify and correct code errors" serve as an example of this procedure in our research. The codes were subsequently integrated and refined into the theme "experience with ChatGPT". To offer additional elucidation, the themes refer to the discernible patterns or concepts that permeate the data and are substantial in delineating a phenomenon while also being relevant to the research inquiry. A theme related to "experience with ChatGPT" may manifest in our research, as student responses inquire about the pros and cons of ChatGPT. Moreover, codes serve as the identifiers assigned to specific data points to concisely represent or summarize the data. For example, situations in which students refer to "extensive programming knowledge" and "various and contextualized responses" could be interpreted as "ChatGPT experience." Moreover, we refer to the data segments that have been assigned unique identifiers as "coded segments." In our research, a coded segment may consist of a participant reminiscing about personal experiences while extracting a sentence or paragraph from the interview transcript.

Fourthly, at the beginning and end of the sessions, we distributed questionnaires to determine the students' perceptions of ChatGPT in the CFP mode. We employed a modified version of the survey instrument created by Venkatesh and Davis (2000) as well as Sánchez and Hueros (2010) for this investigation. Two distinct sections comprised the survey. The primary objective of the initial segment was to gather data from participants concerning their demographic attributes, prior encounters, and anticipations concerning ChatGPT. The following segment comprised questions rated on a seven-point Likert scale, where a rating of one indicated strong disagreement and a rating of seven indicated strong agreement. The investigations revolved around four core domains: perceived usefulness, perceived ease of use, intention to use, and attitude. For additional information, please consult Appendix B. An independent T-test and descriptive analysis were used to ascertain the outcomes based on the gathered data.

Results

Impact of CFP on college students' programming behaviors

Regarding RQ1 (What are the differences in the programming behavior of students engaged in CFP mode compared with those in SDP mode?), we gathered data on students' behaviors during the programming process, and we employed lag-sequential analysis to identify any differences in the behavioral patterns between the two groups. Learners' behavioral patterns exhibited both similarities and discrepancies between the two learning modes. Firstly, in terms of frequency analysis, coding in Python (CP) was the most frequent behavior observed in both classes, followed by either understanding Python code (UPC) or reading feedback (RF). In the SDP mode, the most frequent behaviors were coding in Python (CP; frequency = 5290), reading feedback (RF;

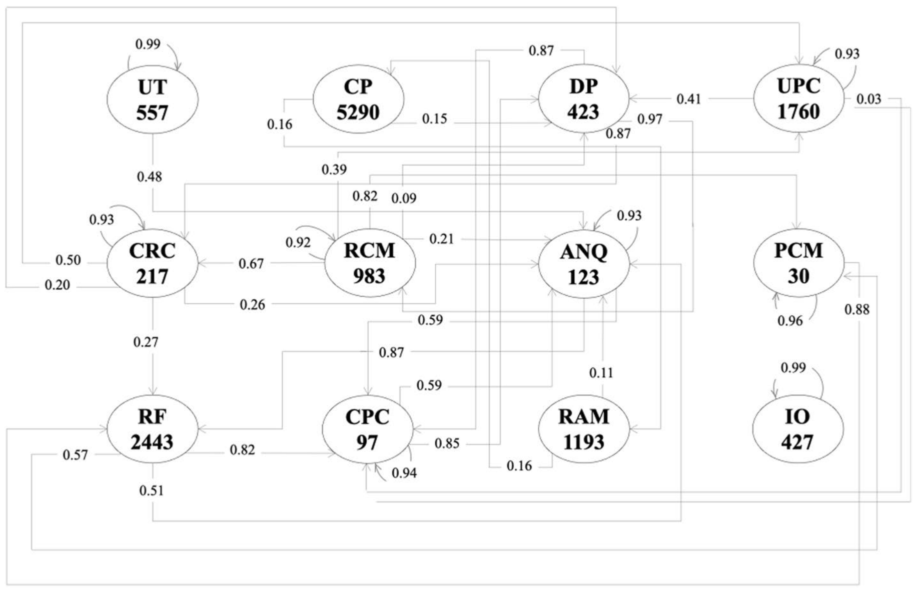


Fig. 2 College students' behavioral sequence diagram in SDP mode, Yule's Q was marked on the line to represent the strength of transitional association. *UT* understanding task, *CP* Coding in Python, *DP* Debugging in Python, *UPC* Understanding Python codes, *CRC* Checking Radar Chart, *RCM* Reading console message, *ANQ* Asking new questions, *PCM* Pasting console message, *PPC* Pasting Python codes, *RF* Reading feedback, *CPC* Copy and paste codes, *RAM* Referring to additional materials, *FC* Failure in ChatGPT, *IO* Idle operation

frequency = 2443), and understanding Python code (UPC; frequency = 1760). Comparatively, learners in the SDP mode engaged in reading feedback behaviors more frequently than those in the CFP mode (RF; frequency = 1439) (see Fig. 2). In the CFP mode, the most frequent behaviors were coding in Python (CP; frequency = 5604), understanding Python code (UPC; frequency = 1779), and reading feedback (RF; frequency = 1439). Conversely, learners in the CFP mode had a higher frequency of code debugging behaviors (DP; frequency = 857) compared to the SDP mode (DP; frequency = 423) (see Fig. 3).

Secondly, regarding sequence analysis, there were 37 and 44 significant programming learning sequences in the SDP and CFP modes, respectively, with numerous links among the different codes. Generally, students in the SDP mode: (1) were more likely to debug Python codes (DP) and receive error messages (RCM) (DP → RCM, Yule's Q = 0.97), or check radar charts (CRC) (DP → CRC, Yule's Q = 0.87); (2) often pasted error messages (PCM) and asked new questions (ANQ) from the output console in the browser to look for solutions (RF) (PCM → RF, Yule's Q = 0.88; ANQ → RF, Yule's Q = 0.87); and (3) preferred to copy and paste codes directly from the browser into their current codes (CPC) and debug to test the correctness of the borrowed codes (DP) (CPC → DP, Yule's Q = 0.85).

As for students in the CFP mode (see Table 2), they (1) frequently copied and pasted codes from ChatGPT (CPC), debugged to test their correctness (DP) (CPC → DP, Yule's Q = 0.95), and then read error messages in the output console window (DP → RCM, Yule's Q = 0.93); (2) preferred to directly copy and paste Python codes (PPC) and error messages from the output console window to ChatGPT (PCM) (PPC → PCM, Yule's Q = 0.86); (3) spent significant time reading feedback in ChatGPT (RF) and copying codes from ChatGPT (CPC) (RF → CPC, Yule's Q = 0.85); and (4) encountered technical

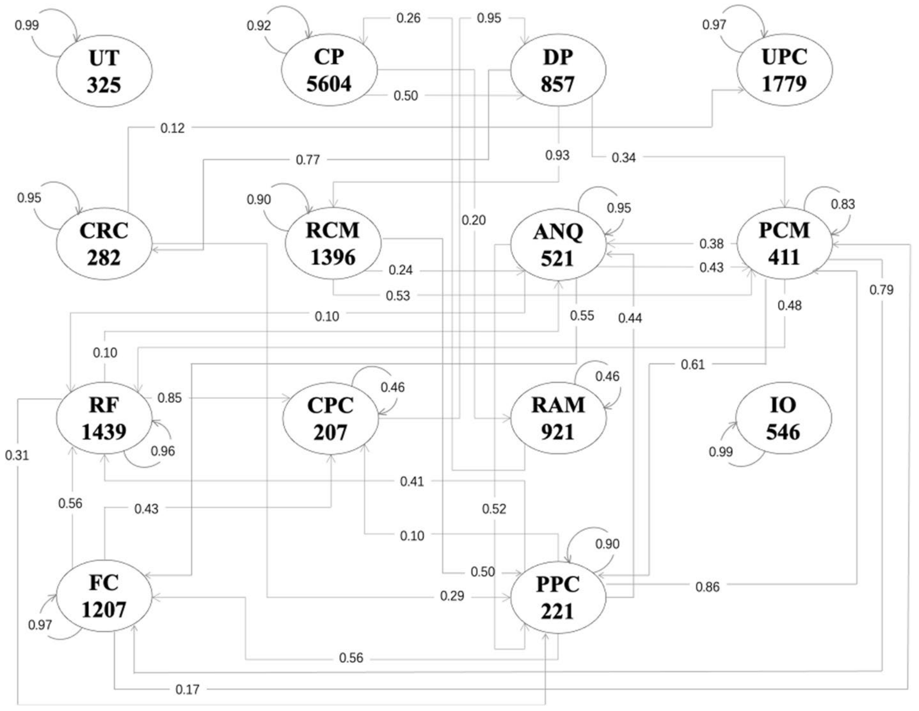


Fig. 3 College students' behavioral sequence diagram in CFP mode

Table 2 LsA transition frequency of programming behaviors of learners in two learning modes

SDP		CFP	
Transition	Yule's Q	Transition	Yule's Q
DP → RCM	0.97	CPC → DP	0.95
PCM → RF	0.88	DP → RCM	0.93
DP → CRC	0.87	PPC → PCM	0.86
ANQ → RF	0.87	RF → CPC	0.85
CPC → DP	0.85	PCM → FC	0.79

Table 3 Independent T-test of college students' programming performance in SDP and CFP modes

Modes	*n*	*M*	*SD*	*t*	*p*
SDP	39	78.36	17.59	− 1.28	0.204
CFP	43	84.11	19.45		

problems (FC) while pasting error messages to ChatGPT (PCM) (FC → PCM, Yule's Q = 0.79). These differences suggest that students in each class employ distinct learning strategies depending on the context and available resources.

Impact of CFP on college students' programming performances

To address RQ2 (What are the differences in the programming performances of students engaged in CFP mode compared with those in SDP mode?), we examined students' programming performance by analyzing their final programming projects, and we presented the results of a T-test that compared the post-test scores between the two groups of learners (see Table 3). After the intervention, learners in the CFP mode had a higher average score (M = 84.11, SD = 19.45) than learners in the SDP mode (M = 78.36, SD = 17.59). A T-test indicated that no statistically significant difference was found between the two classes (t (77) = 1.28, p = 0.204).

College students' user experience about utilizing ChatGPT in CFP practice

With regards to RQ3 (How do college students describe their user experiences with ChatGPT in CFP mode?), we gathered data from students through semi-structured interviews and utilized thematic analysis to explore different themes (Table 4). The services offered by ChatGPT, stages of ChatGPT usage, and experience with ChatGPT were the three main themes that emerged from the thematic analysis of learners' interview data. The first theme explored the services offered by ChatGPT that are most commonly utilized by the students. Thirty students out of a total of forty-three in the CPF group referred to ChatGPT's utilization for programming-related services as advantageous, including error checking, code learning, code writing, code modification, and code interpretation. Wang's remark clarifies that ChatGPT is an invaluable resource for learning and gaining insights into programming concepts and a versatile tool for addressing coding inquiries. In a similar vein, twenty students described how they utilized ChatGPT to perform domain-general tasks, such as browsing for information, synthesizing data, obtaining answers to general inquiries, and identifying unfamiliar concepts.

The second theme highlights the utilization of ChatGPT by students across various stages of the programming process. Among these stages, coding (mentioned by 40 students), debugging (mentioned by 37 students), and acquiring unfamiliar knowledge (mentioned by 25 students) emerged as the top three. Additionally, the students mentioned the importance of decomposing problems (mentioned by 14 students) and

Table 4 Results' of students' user experience on utilizing ChatGPT in CFP practice

Themes	Number of students
Services offered by ChatGPT	
Programming-specific service	40
Domain-general service	19
Stages of ChatGPT-using	
Designing projects	10
Decomposing problems	14
Acquiring unfamiliar knowledges	25
Coding	40
Debugging	37
Experience with ChatGPT	
Expansive programming knowledge	35
Contextualized and varied response	26
Accurate and efficient feedback	15
Human-like interaction	4
Inaccurate codes	17
Limited input and output	10
Technical problems	9

designing projects (mentioned by 10 students) in their interactions with ChatGPT. Yu stated, "Our group used this tool to generate ideas. Afterward, we chose a favorite theme, and ChatGPT helped us expand on it." Meanwhile, Chen highlighted that "ChatGPT's most advantageous feature in programming is its targeted debugging feedback. Copying and pasting error messages into ChatGPT, it helps identify problematic areas in our code."

Among the various experiences of ChatGPT among CPF students, several key points emerged. First of all, the "extensive programming knowledge" from ChatGPT, which includes offering coding solutions and facilitating coding tasks like writing, comprehending, and analyzing code, was mentioned by 35 students. Secondly, the "contextualized and varied responses" provided by ChatGPT, including contextualized feedback and diverse, targeted answers, were mentioned by 26 students. According to students, ChatGPT surpasses traditional search engines like Bing or Baidu by providing not only quick programming solutions but also considering previous queries and offering multiple answers to a given problem. Thirdly, "accurate and efficient feedback" generated from integrating vast resources and saving search time was mentioned by 15 students. Fang found ChatGPT more helpful than popular programming websites like CSDN or GitHub. In contrast to fruitless searches on forums, Fang could input source code or issues into ChatGPT and receive specific and tailored responses, saving significant time. Fourthly, "human-like interaction" was mentioned by 4 students. Yang perceived ChatGPT as a chatbot with a high degree of humanization, stating, "ChatGPT seems to have a human-like understanding of my natural language." On the other hand, Xue pointed out an exceptional aspect of ChatGPT that sets it apart from other tools: its willingness to acknowledge incorrect answers, apologize, and accept suggestions for correction.

However, students also reported several issues, firstly, inaccurate codes arising from un-updated Python libraries, insufficient code compatibility, and inadequate decomposition of large projects were mentioned by 17 students. For example, Wang expressed, "Expecting ChatGPT to generate complete and flawless code for a large programming project is unrealistic." Secondly, "limited input and output" aspects were raised by 10 students. This includes constraints on input and output data types (e.g., mainly text responses), insufficient understanding of "human language", inaccurate comprehension of lengthy conversational information, and demanding requirements for questioning techniques (e.g., prompts). Yang noted, "Formulating questions in ChatGPT can be challenging as it requires logical and clear queries for the desired feedback." Thirdly, technical problems such as instability, slow response times, incomplete feedback (with portions of the question unanswered), and limited comprehension of the Chinese language were mentioned by 9 students.

College students' perceptions towards ChatGPT in CFP practice

Concerning research question 4 (How do college students' perceptions of ChatGPT change following their experience with CFP mode?), we collected student responses via surveys and analyzed the data using T-tests to investigate the students' perspectives on ChatGPT. Using the T-test, this study identified some variations in the perception scores of CFP learners. The mean scores of the students on the pre-test and post-test for the factors comprising four sub-dimensions—"perceived usefulness, perceived ease of use,

Table 5 Statistic results of CFP students' scores on the four dimensions in the pre- and post-test

	Perceived usefulness	Perceived ease of use	Intention to use	Attitude
Pre-test	4.49 (1.41)	4.17 (1.09)	4.60 (1.48)	2.25 (0.81)
Post-test	4.98 (1.08)	4.75 (1.06)	5.22 (0.96)	1.88 (0.57)
t	− 2.34*	− 2.84**	− 3.07**	2.79
Cohen's d	0.46	0.55	0.65	0.56

* $p < 0.05$; ** $p < 0.01$

intention to use, and attitude"—are presented in Table 5. The T-test results indicate that the post-test perceptions of CFP students regarding "perceived usefulness" (t = − 2.34, p = 0.027 < 0.05), "perceived ease of use" (t = − 2.84, p = 0.009 < 0.01), and "intention to use" (t = − 3.07, p = 0.005 < 0.01) were significantly higher than the pre-test perceptions of these variables. However, the statistical analysis did not find a significant difference between the pretest and posttest scores of CFP students in "Attitude" (t = 2.79, p = 0.100 > 0.05).

After the T-test analysis of grade differences, we calculated the effect sizes to assess the significance of the disparities between the pre-test and post-test scores. Researchers often quantify the effect size using Cohen's d value (Cohen, 1988) in the T-test. A d value of 0.2 indicates a small effect size, d = 0.5 indicates a medium effect size and d = 0.8 indicates a large effect size. The findings from Table 4 indicate that the students' evaluations of ChatGPT had a moderate impact on "perceived usefulness" (0.46 > 0.40), "perceived ease of use" (0.55 > 0.50), and "intention to use" (0.65 > 0.50).

According to our analysis, the "perceived usefulness, perceived ease of use, and intention to use" of ChatGPT among college students improved significantly after some time of usage. This finding suggests that students perceived ChatGPT as more practical, user-friendly, and inclined to utilize it again. Conversely, these enhancements failed to yield a favorable influence on their "attitude towards utilization." This implies that while students were receptive to utilizing ChatGPT for practical purposes, the perceived benefits did not result in favorable emotional responses or supersede preexisting inclinations. This highlights a discrepancy between utilitarian and affective considerations. Further investigation is necessary to determine the other factors that influence students' attitudes, in addition to the perception of utility.

Discussions

The primary objective of this research was to investigate the impact of ChatGPT on programming learning among college students. In this study, a comparison was made between two learning modes in terms of their impact on college students' programming learning. The two modes under investigation were instruction with traditional self-directed programming and instruction with ChatGPT-facilitated programming. The objective of this research was to examine and evaluate the disparities in programming performance and programming habits among college students in two distinct learning modalities. Additionally, the study sought to explore the perceptions of these individuals toward ChatGPT programming learning.

Firstly, the results showed that students in each class employed distinct learning strategies depending on the context and available resources. In terms of frequency analysis, coding in Python was the most frequent behavior observed in both classes, followed by either understanding Python code or reading feedback. However, ChatGPT enabled students to paste their original codes or complete error messages, allowing them to receive personalized feedback more easily. This personalized feedback was found to facilitate their programming learning (Sun et al., 2021c). As highlighted by Chen et al. (2023), the improvement in programming learning performance through the use of ChatGPT mostly stems from the provision of code explanations and debugging assistance. In addition to this, Yilmaz and Yilmaz (2023a) noted the features of ChatGPT that facilitate personalized learning, provide clear explanations and programming examples, enable inquiries and searches, and offer resources on advanced topics. Due to Python's popularity, it is clear that hands-on experience and participation in coding activities are crucial to learning the language. By analyzing how college students in a ChatGPT-enabled coding course approach learning, we may better understand how AI can supplement and improve education rather than replace established methods (Wang et al., 2023). Teachers and college students can apply the findings of our research to modify their pedagogical approaches to maximize the utility of tools such as ChatGPT. Still, our study started an important conversation about how to use these AI-powered tools ethically and responsibly. This gives us a chance to look into the best ways to help people learn while still maintaining academic integrity. A concern that emerged during our investigation was the possibility that students might employ ChatGPT or comparable resources inappropriately and cite information from their work. Scholars such as Iqbal et al. (2022) have emphasized the need for transparent communication and explicit protocols regarding the implementation of AI assistance to enhance the learning experience while upholding rigorous academic criteria.

Furthermore, the results of this research indicated that there was no statistically significant distinction between students engaged in the CFP mode and those engaged in the SDP mode. This suggests that the rudimentary implementation of ChatGPT as a facilitator in the programming course does not appear to yield a considerable improvement in student programming performance in comparison to the traditional instructional approach. Contrary to the findings of Yilmaz & Yilmaz et al. (2023a), which suggest that programming instruction aided by ChatGPT and similar tools can enhance students' programming abilities through code explanations and diagnostic assistance, this study's results contradict their findings. In contrast, this study, like Qureshi's (2023), discovered that students who utilized ChatGPT achieved higher scores in programming. However, ChatGPT's code contained errors and contradictions, hindering students from achieving perfect scores in both investigations. The unique contextual elements intrinsic to the study and its results could potentially explain the divergence between the present research findings and those of previous investigations. Other potential influences that could affect the results include the magnitude of the sample, the expertise level of the participants, the quality of instruction, or the specific evaluation methods employed. The results of this study indicate that the utilization of ChatGPT in isolation does not yield a substantial benefit in comparison to conventional self-directed approaches to learning programming. However, proponents argue that the use of AI in schools does not guarantee improved academic performance; instead, it underscores the significance of careful preparation and execution. This research may provide academicians and teachers with insights into the most effective ways to implement AI in the classroom. ChatGPT-facilitated study and self-directed study are two examples that illustrate the significance of comparing and contrasting various teaching approaches. This may assist teachers in assessing novel pedagogical approaches grounded in robust empirical evidence. Thoroughly evaluate the specific context and objectives of the course when integrating AI technology into the classroom.

Thirdly, the thematic analysis of interview data from learners revealed three main themes: the services offered by ChatGPT, the stages of ChatGPT usage, and experience with ChatGPT. The services offered by ChatGPT include programming-specific services and domain-general services. The stages of ChatGPT usage include designing projects, decomposing problems, acquiring unfamiliar knowledge, coding, and debugging. The advantages of using ChatGPT include having extensive programming knowledge, contextualized and varied responses, and accurate and efficient feedback. Other studies have also advocated for the advantages of ChatGPT, such as providing rapid and accurate responses, availability at any time and location, ease of access, utilizing natural language, facilitating personalized learning, supporting multiple languages, offering clear explanations and programming examples, as well as providing resources for advanced topics. Besides, the study identified several disadvantages of using ChatGPT, including inaccurate codes, limited input and output, and technical problems. Other studies have also identified additional disadvantages, such as causing occupational anxiety, providing inadequate information, lacking a real programming environment, and potentially increasing laziness among programmers (Yilmaz & Yilmaz, 2023a).

CFP students held favorable views regarding the implementation of ChatGPT. More precisely, we observed significant improvements in the perceived usefulness, perceived ease of use, and intention to use. The effect size of the observed enhancements was moderate, indicating that there was a noticeable influence. Although the findings fail to demonstrate a statistically significant improvement in their inclination towards utilizing ChatGPT, To summarize, the implementation of the intervention led to enhanced attitudes and intentions towards the utilization of ChatGPT. Nevertheless, this did not yield a substantial influence on the perception of utilizing ChatGPT. According to the findings, it is evident that students' perceptions of the technology or intervention's efficacy, usability, and propensity to employ it have all improved considerably since its implementation. Nevertheless, their viewpoint regarding the criticality of technology utilization remained essentially unchanged. The outcomes of this research corroborate the findings of Elkhodr et al. (2023), which indicated that students held a positive view of ChatGPT, regarding it as a beneficial and enjoyable educational resource. A significant proportion of students demonstrated a predilection to employ these artificial intelligence methods in future occurrences. Moreover, the study revealed that students who utilized ChatGPT exhibited improved performance with regard to functionality, user flow, and comprehension of the material, as opposed to those who exclusively relied on traditional search engines. Furthermore, the findings of this research corroborate those of Limna et al. (2023), which reported that the perception of ChatGPT's implementation in educational settings is predominantly positive among both instructors and learners. On the basis of the results obtained, it is possible to deduce that the intervention produced positive outcomes with respect to the students' attitudes and intentions towards the technology. However, the impact of the intervention on their attitudes was negligible. The propensity of the students to adopt technology can be attributed to the benefits they perceive as well as the technology's user-friendly characteristics. However, the lack of a significant change in perspective indicates that there may still be reservations or contradictory views among individuals concerning the execution of this endeavor. In general, the results indicate that college students perceive ChatGPT as advantageous and intuitive and are inclined to integrate it into their forthcoming academic and vocational pursuits. This study highlights the importance of comprehending the potential advantages and disadvantages of integrating AI language models into programming education, as well as the potential impact of these technologies on the perspectives and attitudes of individuals, including teachers and learners, towards technology.

Implications

Considering the research findings, this study proposes pedagogical and developmental suggestions for the future integration and implementation of ChatGPT. To begin with, teachers ought to maintain an awareness of the potential merits and demerits associated with the incorporation of ChatGPT or other AI language models into their computer programming courses from an instructional standpoint. While ChatGPT can provide benefits like personalized feedback and contextualized responses, it can also lead to inaccuracies and inconsistencies in code (Chen et al., 2023). Therefore, instructors should plan and implement AI-based educational resources and systems carefully and thoughtfully. For instance, instructors could enhance the material's understanding among students by providing clear explanations and programming examples of utilizing ChatGPT. A "prompt" establishes the context for a task or corpus of text that the language model completes following AI technology. Constructing an appropriate prompt is crucial as the model strives to generate text that aligns with the context established by the initial prompt (Liu et al., 2023; Reynolds & McDonell, 2021).

In addition, during the ChatGPT-facilitated programming process, instructors need to integrate effective strategies for utilizing ChatGPT, and they should also monitor students closely for instances of academic dishonesty, such as plagiarism when using ChatGPT or other web resources. On the development level, the accuracy and consistency of the model's generated code are the top priorities. As highlighted by this research study and previous studies (Jalil et al., 2023; Yilmaz & Yilmaz, 2023b), inaccuracies in the code generated by ChatGPT limit its effectiveness and may lead to instances of academic dishonesty such as plagiarism. By improving the accuracy and consistency of the code generated, ChatGPT can become a more reliable and valuable tool for programming learning. Additionally, developers can also consider integrating a framework for heuristic guidance in ChatGPT that is modeled after Socrates' method of questioning. Rather than providing direct answers to programming problems, this framework should guide students toward solving problems by asking sequential and probing questions. By encouraging students to think creatively and critically, the heuristic guidance framework can help enhance their problem-solving skills. Additionally, this framework has the potential to provide students with a more personalized learning experience by streamlining the process of comprehending programming fundamentals and their practical implementation.

Conclusion, limitations, and future directions

ChatGPT is an advanced AI language model that possesses the potential to assist in computer programming endeavors by providing guidance, answering inquiries, and generating code snippets. This quasi-experimental research compared college students' programming performances and behaviors between self-directed programming and ChatGPT-facilitated programming modes, investigated students' experiences and perceptions of utilizing ChatGPT during programming and revealed critical discrepancies between the two learning modes. Integrating and implementing generative AI tools, such as ChatGPT, in programming curricula is certainly possible. Nevertheless, it is critical to recognize the limitations that are inherent in the scope of this study. First, the findings may have limited generalizability due to using a specific version of ChatGPT; future studies should compare results across multiple versions or models. Second, the current study aimed to examine how students utilize ChatGPT as a tool for programming education in authentic learning contexts; however, considering the significance of instructional prompts and similar learning tools, future research endeavors will delve into instructional strategy design, particularly concerning prompt use strategies. Thirdly, the fact that the duration of the research was limited to five weeks emphasizes the need for longitudinal investigations that span longer periods to gauge students' perspectives on ChatGPT utilization.

The incorporation of ChatGPT into programming instruction significantly enhances students' programming studies, as indicated by the findings of the present study. Despite recognizing various constraints in this investigation, it is necessary to conduct further scholarly inquiry to build upon these discoveries and explore the full potential of AI language models in enhancing programming instruction. Therefore, it is justifiable to assert that ChatGPT and analogous AI tools have the capacity to function as powerful tools for teachers and students alike, augmenting the pedagogical experience in programming.

Appendix A

Interview protocol

Do you like using ChatGPT and why?

What services does ChatGPT offer you?

What do you think about using ChatGPT in this course so far?

At what stages of program design do you usually use ChatGPT?

How do you use ChatGPT during the problem-solving process in programming?

What do you think is good about ChatGPT?

What are the biggest problems you've encountered with ChatGPT?

Appendix B

Survey for the perception of ChatGPT

Name.

Sex.

How well are you currently using ChatGPT?

Perceived usefulness

ChatGPT helps me to learn programming more efficiently.

ChatGPT improves my programming performance.

ChatGPT makes my learning more effective.

ChatGPT makes it easier to learn programming.

Overall, ChatGPT is advantageous for my programming learning.

Perceived ease of use

It is easy to get materials from ChatGPT.

The process of using ChatGPT is clear and understandable.

Overall, I believe that ChatGPT is easy to use.

Intention to use

I will use ChatGPT in the future.

I am willing to try ChatGPT in a variety of areas, such as copywriting, multimedia creation, recreation, and other work and life situations.

Attitude

Learning on ChatGPT is fun.

Using ChatGPT is a good idea.

ChatGPT is an attractive way to learn.

Overall, I like using ChatGPT.

Acknowledgements
The authors express gratitude towards all the participants for their enthusiastic involvement and cooperation in this study.

Author contributions
Dan Sun conceptualized the study, designed the methodology to conduct formal analysis, and wrote the original draft. Azzeddine Boudouaia wrote the original draft and conducted data analysis. Chengcong Zhu conducted data collection and analysis. Yan Li proofread the manuscript. All authors revised the draft and contributed to the final manuscript.

Funding
This work was supported by funding from the National Natural Science Foundation of China [Grant No. 62307011] and the Zhejiang University [Grant No. S20230157] and Graduate Education Research Center of Zhejiang University [Grant No. YJSJY20240101].

Availability of data and materials
Data will be made available on request.

Declarations

Competing interests
There is no competing interest to declare.

Received: 14 September 2023 Accepted: 29 January 2024
Published online: 22 February 2024

References

Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM, 60*(6), 72–80. https://doi.org/10.48550/arXiv.1705.09413

Chen, B., Resendes, M., Chai, C. S., & Hong, H. Y. (2017). Two tales of time: Uncovering the significance of sequential patterns among contribution types in knowledge-building discourse. *Interactive Learning Environments, 25*(2), 162–175. https://doi.org/10.1080/10494820.2016.1276081

Chen, E., Huang, R., Chen, H. S., Tseng, Y. H., & Li, L. Y. (2023). *GPTutor: A ChatGPTpowered programming tool for code explanation*. arXiv preprint arXiv:2305.01863.

Cheng, G. (2019). Exploring factors influencing the acceptance of visual programming environment among boys and girls in primary schools. *Computers in Human Behavior, 92*, 361–372. https://doi.org/10.1016/j.chb.2018.11.043

Chevalier, M., Giang, C., Piatti, A., & Mondada, F. (2020). Fostering computational thinking through educational robotics: A model for creative computational problem solving. *International Journal of STEM Education, 7*(1), 1–18. https://doi.org/10.1186/s40594-020-00238-z

Cohen, J. (1988). *The effect size* (pp. 77–83). Routledge.

Cohen, L., Manion, L., & Morrison, K. (2013). *Research methods in education*. Routledge

Davis, F. D. (1989). Perceived usefulness, perceived ease of use and user acceptance of information technology. *MIS Quartely, 13*(3), 319–340.

Elkhodr, M., Gide, E., Wu, R., & Darwish, O. (2023). ICT students' perceptions towards chatgpt: An experimental reflective lab analysis. *STEM Education, 3*(2), 70–88. https://doi.org/10.3934/steme.2023006

Faraone, S. V., & Dorfman, D. D. (1987). Lag sequential analysis: Robust statistical methods. *Psychological Bulletin, 101*(2), 312–323. https://doi.org/10.1037/0033-2909.101.2.312

Firaina, R., & Sulisworo, D. (2023). Exploring the usage of ChatGPT in higher education: Frequency and impact on productivity. *Buletin Edukasi Indonesia, 2*(01), 39–46. https://doi.org/10.56741/bei.v2i01.310

Gervasi, O., Murgante, B., Misra, S., Garau, C., Blečić, I., Taniar, D., Apduhan, B. O., Rocha, A. C., Tarantino, A. C., & E., & Torre, C. M. (2021). Inference engines performance in reasoning tasks for intelligent tutoring systems. *Computational science and its applications* (pp. 471–482). Springer International Publishing AG. https://doi.org/10.1007/978-3-030-86960-1_33

Ghatrifi, M. O., Amairi, J. S., & Thottoli, M. M. (2023). Surfing the technology wave: An international perspective on enhancing teaching and learning in accounting. *Computers and Education. Artificial Intelligence, 4*, 100144. https://doi.org/10.1016/j.caeai.2023.100144

ICOURSE. retrieved from 28 July, 2023 from https://www.icourse163.org/course/BIT-268001?from=searchPage&outVendor=zw_mooc_pcssjg_

Iqbal, N., Ahmed, H., & Azhar, K. A. (2022). Exploring teachers' attitudes towards using Chat GPT. *Global Journal for Management and Administrative Sciences, 3*(4), 97–111. https://doi.org/10.46568/gjmas.v3i4.163

Jalil, S., Rafi, S., LaToza, T. D., Moran, K., & Lam, W. (2023). Chatgpt and software testing education: Promises & perils. In *2023 IEEE international conference on software testing, verification and validation workshops (ICSTW)* (pp. 4130–4137). IEEE.

Jancheski, M. (2017). Improving teaching and learning computer programming in schools through educational software. *Olympiads in Informatics, 11*(1), 55–75. https://doi.org/10.15388/ioi.2017.05

Javaid, M., Haleem, A., Singh, R. P., Khan, S., & Khan, I. H. (2023). Unlocking the opportunities through ChatGPT tool towards ameliorating the education system. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations, 3*(2), 100115. https://doi.org/10.1016/j.tbench.2023.100115

Khan, I., Al Sadiri, A., Ahmad, A. R., & Jabeur, N. (2019). Tracking student performance in introductory programming by means of machine learning. In *Proceedings of MEC international conference on big data and smart city (ICBDSC)* (pp.1–6).

Klasnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z., & Jain, L. C. (2016). *Case study: Design and implementation of programming tutoring system*. Springer International Publishing AG.

Limna, P., Kraiwanit, T., Jangjarat, K., Klayklung, P., & Chocksathaporn, P. (2023). The use of chatgpt in the digital era: Perspectives on Chatbot Implementation. *Journal of Applied Learning & Teaching, 6*(1), 64–74. https://doi.org/10.37074/jalt.2023.6.1.32

Lo, C. K. (2023). What is the impact of chatgpt on education? A Rapid Review of the literature. *Education Sciences, 13*(4), 410. https://doi.org/10.3390/educsci13040410

Looi, C. K., How, M. L., Wu, L. K., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255–279. https://doi.org/10.1080/08993408.2018.1533297

Lu, O. H. T., Huang, J. C. H., Huang, A. Y. Q., & Yang, S. J. H. (2017). Applying learning analytics for improving students engagement and learning outcomes in an MOOCs enabled collaborative programming course. *Interactive Learning Environments, 25*(2), 220–234. https://doi.org/10.1080/10494820.2016.1278391

Malik, S. I., Ashfque, M. W., Tawafak, R. M., Al-Farsi, G., Ahmad Usmani, N., & Hamza Khudayer, B. (2022). A chatbot to facilitate student learning in a programming 1 course: A gendered analysis. *International Journal of Virtual and Personal Learning Environments, 12*(1), 1–20. https://doi.org/10.4018/IJVPLE.310007

Mousavinasab, E., Zarifsanaiey, N., NiakanKalhori, R. S., Rakhshan, M., Keikha, L., & Ghazi Saeedi, M. (2021). Intelligent tutoring systems: A systematic review of characteristics, applications, and evaluation methods. *Interactive Learning Environments, 29*(1), 142–163. https://doi.org/10.1080/10494820.2018.1558257

Nurbekova, Z., Tolganbaiuly, T., Nurbekov, B., Sagimbayeva, A., & Kazhiakparova, Z. (2020). Project-based learning technology: An example in programming microcontrollers. *International Journal of Emerging Technologies in Learning, 15*(11), 218–227. https://doi.org/10.3991/ijet.v15i11.13267

OpenAI. (2023). *Introducing ChatGPT*. Retrieved July 30, from https://openai.com/blog/chatgpt

Peng, R., Hu, Q., & Kouider, B. (2023). Teachers' acceptance of online teaching and emotional labor in the EFL context. *Sustainability, 15*(18), 13893–13893. https://doi.org/10.3390/su151813893

Qureshi, B. (2023). *Exploring the use of chatgpt as a tool for learning and assessment in undergraduate computer science curriculum: Opportunities and challenges.* arXiv preprint arXiv:2304.11214.

Rahman, M. M., & Watanobe, Y. (2023). ChatGPT for education and research: Opportunities, threats, and strategies. *Applied Sciences, 13*(9), 5783.

Ray, B., Posnett, D., Filkov, V., & Devanbu, P. (2014). A large scale study of programming languages and code quality in github. In *Proceedings of the 22nd ACM SIGSOFT international symposium on foundations of software engineering* (pp. 155–165).

Sánchez, R. A., & Hueros, A. D. (2010). Motivational factors that influence the acceptance of Moodle using TAM. *Computers in Human Behavior, 26*(6), 1632–1640. https://doi.org/10.1016/j.chb.2010.06.011

Shen, G., Yang, S., Huang, Z., Yu, Y., & Li, X. (2023). The prediction of programming performance using student profiles. *Education and Information Technologies, 28*(1), 725–740. https://doi.org/10.1007/s10639-022-11146-w

Sivasakthi, M. (2017). Classification and prediction based data mining algorithms to predict students' introductory programming performance. In *Proceedings of international conference on inventive computing and informatics (ICICI)* (pp. 346–350).

Skalka, J., Drlik, M., Benko, L., Kapusta, J., Pino, D., Rodríguez, J. C., Smyrnova-Trybulska, E., Stolinska, A., Svec, P., & Turcinek, P. (2021). Conceptual framework for programming skills development based on microlearning and automated source code evaluation in virtual learning environment. *Sustainability, 13*(6), 3293. https://doi.org/10.3390/su13063293

Stehle, S. M., & Peters-Burton, E. E. (2019). Developing student 21st century skills in selected exemplary inclusive STEM high schools. *International Journal of STEM Education, 6*(1), 1–15. https://doi.org/10.1186/s40594-019-0192-1
Sun, D., Ouyang, F., Li, Y., & Zhu, C. (2021b). Comparing learners' knowledge, behaviors, and attitudes between two instructional modes of computer programming in secondary education. *International Journal of STEM Education, 8*(1), 54–54. https://doi.org/10.1186/s40594-021-00311-1
Sun, D., Ouyang, F., Li, Y., & Chen, H. (2021c). Three contrasting pairs' collaborative programming processes in China's secondary education. *Journal of Educational Computing Research, 59*(4), 740–762. https://doi.org/10.1177/0735633120973430
Sun, L., Hu, L., & Zhou, D. (2021d). Which way of design programming activities is more effective to promote K-12 students' computational thinking skills?A meta-analysis. *Journal of Computer Assisted Learning*. https://doi.org/10.1111/jcal.12545
Sun, D., Xu, F., & Li, Y. (2021a). Using learning analytics in understanding students' behavior in block-based and text-based programming modality. In *2021 tenth international conference of educational innovation through technology (EITT)*. https://doi.org/10.1109/eitt53287.2021.00060
Surameery, N. M. S., & Shakor, M. Y. (2023). Use chatgpt to solve programming bugs. *International Journal of Information Technology & Computer Engineering (IJITC), 3*(1), 17–22.
Teo, T., Luan, W. S., & Sing, C. C. (2008). A cross-cultural examination of the intention to use technology between Singaporean and Malaysian pre-service teachers: An application of the technology acceptance model (TAM). *Educational Technology and Society, 11*(4), 265–280.
Thongkoo, K., Daungcharone, K., & Thanyaphongphat, J. (2020). Students' acceptance of digital learning tools in programming education course using technology acceptance model. *IEEE Xplore*. https://doi.org/10.1109/ECTIDAMTNCON48261.2020.9090771
Tian, H., Lu, W., Li, T. O., Tang, X., Cheung, S. C., Klein, J., & Bissyand ́e, T. F. (2023). *Is ChatGPT the ultimate programming an assistant–How far is it?*. arXiv preprint arXiv: 2304.11938.
Tlili, A., Shehata, B., Adarkwah, M. A., Bozkurt, A., Hickey, D. T., Huang, R., & Agyemang, B. (2023). What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart Learning Environments, 10*(1), 15.
Tom, M. (2015). Five cs framework: A student-centered approach for teaching programming courses to students with diverse disciplinary background. *Journal of Learning Design, 8*(1), 21–27. https://doi.org/10.5204/jld.v8i1.193
Venkatesh, V., & Davis, F. D. (2000). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science, 46*(2), 186–204. https://doi.org/10.1287/mnsc.46.2.186.11926
Wang, H., Tlili, A., Huang, R., Cai, Z., Li, M., Cheng, Z., Yang, D., Li, M., Zhu, X., & Fei, C. (2023). Examining the applications of intelligent tutoring systems in real educational contexts: A systematic literature review from the social experiment perspective. *Education and Information Technologies, 28*(7), 9113–9148. https://doi.org/10.1007/s10639-022-11555-x
Xie, Y., Boudouaia, A., Xu, J., AL-Qadri, A. H., Khattala, A., Li, Y., & Aung, Y. M. (2023). A study on teachers' continuance intention to use technology in English instruction in western China junior secondary schools. *Sustainability, 15*(5), 4307. https://doi.org/10.3390/su15054307
Yang, F.-Y., & Tsai, C.-C. (2008). Investigating university student preferences and beliefs about learning in the web-based context. *Computers and Education, 50*, 1284–1303.
Yilmaz, R., & Yilmaz, F. G. K. (2023a). Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning. *Computers in Human Behavior: Artificial Humans, 1*(2), 100005. https://doi.org/10.1016/j.chbah.2023.100005
Yilmaz, R., & Yilmaz, F. G. K. (2023b). The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education. Artificial Intelligence, 4*, 100147. https://doi.org/10.1016/j.caeai.2023.100147
Yilmaz, R., & Yilmaz, F. G. K. (2023c). The effect of generative artificial intelligence (AI) based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education: Artificial Intelligence, 4*, 100147.

Publisher's Note

Dan Sun (Ph.D.) is an assistant professor in Jing Hengyi School of Education at Hangzhou Normal University. Her research interests include artificial intelligence in education, programming education, learning analytics and computational thinking, etc.

Azzeddine Boudouaia (Ph.D.) is a postdoctoral fellow in educational technology at the College of Education, Zhejiang University, PR China. His research interests include curriculum studies, technology and artificial intelligence in EFL education, EFL teaching approaches, and teacher professional development.

Chengcong Zhu is a high school teacher in Xiaoshan High School. His research interests include information technology, programming education, etc.

Yan Li (Ph.D.) is a professor in College of Education at Zhejiang University. Her research interests include e-learning, distance education, ICT education, etc.