Who Seeks AI Help? Student Characteristics and Performance Outcomes with LLM-based Programming Assistant

Neha Rani [1,*], Anh Le [1], Joanna M. Mijares [2], and Ashish Aggarwal [3]

[1] Department of Computer and Information Science and Engineering, University of Florida, Gainesville, USA
[2] College of Liberal Arts and Sciences, University of Florida, Gainesville, USA
[3] Department of Engineering Education, University of Florida, Gainesville, USA
Email: neharani@ufl.edu (N.R.); vu.le@ufl.edu (A.L.); jmijares@ufl.edu (J.M.M.), ashishjuit@ufl.edu (A.A.)
*Corresponding author

Manuscript received April 1, 2026; revised May 22, 2026; accepted July 30, 2026; published November 19, 2026

***Abstract*—With the rising popularity of computer science programs, students from different majors are enrolling in programming courses. There is a lack of personalized feedback and one-on-one guidance for learning programming due to large class sizes. This study explores the potential of the Large Language Model (LLM)-based application, CodeHelp, for learning programming in introductory programming courses. CodeHelp is an Artificial Intelligence (AI) tool that lets students submit problematic code snippets, error messages, and questions, offering guidance without giving direct answers. The study aims to identify the characteristics of students who utilized CodeHelp and how it impacted their performance in the course. CodeHelp was provided in an introductory programming class at a university in the United States with 508 students enrolled. A total of 155 students chose to explore CodeHelp. Survey responses, CodeHelp usage, and course outcomes were used for analysis. Results show that CodeHelp users are students with better time management skills and who report a higher sense of responsibility and accountability compared to non-users. CodeHelp users also showed improved academic performance compared to non-users.**

***Keywords*—Artificial Intelligence (AI)-tutor, Large Language Models (LLMs), introductory programming, learning programming, AI programming assistant, CodeHelp**

I. Introduction

As the popularity of computer science grows, more and more students are enrolling in computer science programs and courses, particularly programming courses [1]. Developing programming skills is one of the key objectives for students enrolled in undergraduate computer science programs [1]. As the number of students learning programming continues to grow, there is an increase in the demand for effective programming assistance tools. Such tools can support students learning to program who come from varied backgrounds, demographics, and levels of experience. Many students, especially those who are new to programming, struggle with learning and understanding fundamental concepts and solving coding challenges independently due to large class sizes [2, 3]. Large classes lead to reduced instructor attention towards individual students and a lack of personalized assistance and feedback [4]. In addition, some students may feel uncomfortable seeking help from instructors and teaching assistants or requesting in-person guidance because of the fear of being judged. They may be hesitant to ask basic questions, leading to missed opportunities for learning. These difficulties are compounded by the nature of classrooms where one instructor is often responsible for many, possibly hundreds of students. This setting makes it nearly impossible for the instructor to provide timely, personalized feedback to each student. Besides, coding assignments often present varying degrees of difficulty; one student may struggle with basic syntax understanding, and another may need help with complex algorithmic problems. Further, each individual student may bring a level of expertise and experience and face different levels and types of difficulties that require more personalized attention. This range of challenges demands an adaptable solution, which is lacking in a traditional one-size-fits-all approach used in classrooms.

In response to these challenges, the rise of Generative Artificial Intelligence (AI) based programming tools in the educational environment has gained significant attention [5]. AI has many possible useful applications in classroom settings, enhancing both teaching and learning experiences. It presents the opportunity to provide solutions to the lack of personalized assistance and feedback in classroom settings. These tools can help students progress at their own pace, assist instructors, and potentially reduce administrative workload [6, 7]. AI use in the classroom has become nearly commonplace, whether it is officially permitted or not. Generative AI-based tools can support learning by providing personalized feedback [8]. Further, AI also contextualizes information using users' prior queries and interactions. This contextualization further aids learning by tailoring responses according to users' prior knowledge and interactions [9, 10]. While there are opportunities, there are also some challenges associated with using AI that should be accounted for. Chatbots may provide writing samples or even code blocks upon request, so it is not uncommon for students to take these samples provided and plagiarize them or use them as references. This may create concerns for lower engagement with the learning task. There are also concerns of plagiarism with the use of AI. Additionally, detecting AI use in homework and assignments can be particularly challenging. Further, due to the generative nature of AI-based conversational chatbots, they can hallucinate, leading to incorrect solutions. If the learner does not understand the problem-solving approach and rely on AI, they might not notice AI inaccuracies and form false conceptions. Further, even if they are correct, copying and pasting responses will not help students understand the concept and absorb knowledge. Instead, it can raise concerns about academic integrity. Further, frequent use of AI can lead to overreliance and compromised learning [11]. Overreliance can lead not only to the formation of incorrect knowledge when AI provides an incorrect response, but also to turning to AI for

every task. Resorting to AI for educational tasks further leads to cognitive offloading and may affect individual cognitive skills [12].

To address the need for more personalized assistance for a large number of students and to explore the potential of AI in classrooms, CodeHelp was created by Mark Liffiton and developed by Liffiton *et al*. [13]. CodeHelp is an open-source project and was made publicly available in August 2023. It makes use of OpenAI's "GPT-3.5" in order to generate responses for students [14]. Prior work has pointed towards having more guardrails in AI-based technologies when used in an educational context, which is likely to make them suitable for educational use [15]. Unlike traditional programming assistance systems and other AI tools, CodeHelp stands out in this domain by offering targeted guidance without providing explicit answers. This unique characteristic encourages students to have independent problem-solving skills by exploring solutions on their own with guidance rather than directly using given solutions. It also helps somewhat in dealing with inaccuracies in AI output, as it does not provide a direct solution but explains the problem better and guides students in the right direction. Thus, encouraging critical thinking and a self-dependent approach to problem-solving will enable learners to be more aware of potential inaccuracies in AI outputs. This allows students to get appropriate help when they seek help and need extra scaffolding to learn better. For students who are struggling or new to programming, CodeHelp assists them by prompting them to think about the right approach. This method fosters a deeper understanding, which is needed for new programmers to form their own mental models of the programming concepts. This can also assist with self-regulated learning and allow learners to learn at their own pace and level of depth on a topic during out-of-class exploration. Self-regulated learning is often guided by individual disposition to learn, their interests, self-control, and metacognition [16]. Therefore, different students may have different usage and usefulness perceptions of the tool. When using CodeHelp, the software provides students with the option to input which programming language they are working in to give the AI model a better understanding of the issue as a whole and directions for solution without directly handing a solution. CodeHelp can help with self-reflection of problems and concepts and therefore help with self-regulated learning. This paper presents findings of deploying CodeHelp, an AI-based conversation programming assistant tool with guardrails, in a real classroom setting in a large-enrollment fundamental programming course. CodeHelp was deployed in an Introductory Programming classroom of 508 students during the Spring 2024 semester within an undergraduate computer science program at a higher education university in the United States of America. CodeHelp was made available to the class for approximately 15 weeks (about 3 and a half months), and using the software was optional for students in the course. A survey was sent at the start of the semester to gauge student confidence, abilities, and demographics. A second survey was sent out at the very end of the semester, asking about their experience in the course, their grades, and whether they used CodeHelp. The survey included both multiple choice questions and open-ended responses. Of the 508 students, 155 submitted at least one query to CodeHelp. Our study aims to explore whether CodeHelp effectively supports students' programming learning journey, examining not only whether it helps its users but also how demographically diverse student groups use it. The Research Questions (RQs) guiding this study are:

RQ1: Does the use of Generative AI-based programming assistance tools, such as CodeHelp, support students' academic performance?

RQ2: What are the characteristics of students who use CodeHelp versus those who do not?

This paper is structured as follows. The next section reviews and presents the related work, examining various AI-based programming assistance tools and their impact on learning outcomes, as well as traditional non-AI programming help tools. Section IV outlines the data collection process, detailing how logs were collected and surveys conducted to gather information about users' demographics, their experiences with CodeHelp, the course's outcome, and their interactions with the tool. Section V focuses on participants, providing insight into sample size and participants' characteristics. Section VI, describes the analysis method used, including data filtering and the statistical tests applied. Finally, Section VII presents the findings, highlighting the results of our analyses before concluding the paper.

II. Background and Related Work

The rapid growth of the field of AI in recent decades has led to a multitude of different uses for AI and Large Language Models (LLMs), and there exists potential for AI to be used in multiple aspects of education [8, 12, 17–19]. AI, as we know it today, is the product of many years of development, and LLMs are a large part of that development. LLMs are designed to "understand" human language and create outputs based on that understanding [20]. Large amounts of data and text are input into these LLMs to train them and accustom them to human speech patterns and language, and outputs are created by the LLMs after training and analysis are completed. LLMs do not technically "understand" speech as a human would, but they can replicate this understanding via analysis of patterns [20, 21].

Large Language Models are the focal point of "chat" software like ChatGPT and Google Gemini. However, these LLMs are not without fault [22]. Any bias introduced into training material has been shown to produce biased responses from these LLMs, and there is a slight risk of incorrect responses if not enough data is provided to the LLM on a particular subject [20]. Further, if the LLM has not used recent information for training, it might not provide the most relevant response. Given the recent exponential improvements in LLMs, there has been growing interest in AI across a wide range of fields, including computing education. They now offer human-like natural language responses with high accuracy.

Both formal and informal education, in general, is observing a surge in the use of AI technologies [23, 24]. It has wide-ranging usage, such as personal exploration of a topic of interest, addressing individual curiosity, assisting in learning coding, medical training education, and in-class usage [19, 25, 26]. In an investigation based on interviews

with 20 programming instructors, Lau and Gau [17] found that due to the initial skepticism and plagiarism issue, many educators discouraged the use of AI-assisted tools and increased exam invigilation. This was an initial response to AI in education. However, recently, more and more educators have begun to embrace AI tools and acknowledge its undeniable presence, moving towards acceptance. This has led to greater awareness of AI use cases and the creation of AI with guardrails for the appropriate use of AI assistance tools. Some have begun to integrate AI-based tools in their classes [17, 19]. Modern use of AI in the average classroom may look like assisting teachers with teaching repetitive subjects or assisting with lesson plans and personalized assistance [26, 27].

The use of AI-driven tools in programming education, in particular, has grown in response to the demand for learning more advanced programming subjects for learners with varied backgrounds and experiences and the need to cater to individual needs [28]. These tools not only support instructors by offering resources that streamline teaching programming but also allow students to receive instant feedback and guidance on their code at any point at their own pace [29]. This personalization plays a crucial role in encouraging students who struggle with traditional, generalized instruction methods to engage more actively with learning content. Prior studies have shown that AI tools such as ChatGPT significantly enhance students' critical thinking, engagement, programming self-efficacy, and motivation and support self-regulated learning [19, 23, 29]. Students feel more confident expanding on concepts from the lectures when they can quickly access code solutions and explanations. In addition to ChatGPT, AI-powered tools such as GitHub Copilot have also been widely adopted in educational settings [26, 30]. Copilot helps developers quickly look at code samples, identify bugs, and even refactor code [31].

A recent investigation explored the use of GitHub Copilot in software engineering upper-division courses to add features to a large codebase, examining both Copilot chat and code-generation features [30]. Exploring how students use these chat-based tools is essential, as it helps understand the usage patterns and perceptions. Especially when these skills are actively used by working software professionals, and the university-level students are the ones who will join the workforce [30]. Another AI tool, Visualized Object-Oriented Programming Learning Tool (VTL-OOP), is designed to enhance students' motivation and performance by offering a visual programming approach as an alternative to the traditional text-based programming approach [32]. This visual approach makes programming more accessible to beginners and fosters a deeper understanding of programming concepts. In another recent work, a tool with guardrails for learning programming called CodeAid has been developed and investigated [33]. CodeAid is an LLM-based programming assistance tool that provides technical responses without providing direct answers. It produces pseudo code and a line-by-line explanation of a code block. It also annotates bugs in students' code and suggests fixes. This investigation focused on gathering design considerations for AI assistant tools, but it did not explore who the students are who use them.

Despite the popularity of AI tools, the literature suggests that the use of AI in programming education is still in its early stages, with limited studies on its effectiveness, usage pattern, its impacts, and relating usage to user characteristics [15]. There are some concerns by instructors and educators that students may potentially rely heavily on generated responses, potentially even plugging in assignments and pasting entire blocks of code or text [34]. This new avenue of generating an entire project's worth of work is a concern for academic integrity, but the potential for plagiarism in writing and programming has always existed in the academic sphere [15].

The concept of guiding students to work through the problem using AI, rather than supplying them with answers, has much potential in this field. This version of AI-based solutions for assistance in education forms a way for acceptable use of AI with guardrails that support learning without raising concerns for lack of cognitive development, active engagement in education, and critical thinking. CodeHelp is a tool that maneuvers students towards answers and understanding rather than handing out solutions. Previous work and research on the use of CodeHelp in classrooms has also been conducted. One study reported that students sought CodeHelp for immediate assistance with errors more than to learn about concepts [35]. They also reported that students tend to provide short instructions to CodeHelp, indicating a need for training on prompt formation [35]. Another study was conducted on a first-year computer science and data science course of 52 students over 12 weeks [13]. This study [13] was facilitated by the creators of CodeHelp. The average number of queries submitted per student was 40. Through the optional survey at the end of that course, students reported that CodeHelp was valuable and useful. Over 95% of students stated they would be willing to use CodeHelp in the future. Further analysis of the patterns found between CodeHelp usage, survey responses, and course data indicates a positive relationship between tool usage and course performance [14]. However, this study was conducted with a very small class size, which may not be representative of the collective student population, especially in a large classroom setting where the importance of such tools may increase. Therefore, an investigation with a larger, more diverse classroom of 508 students to investigate the usage of a broader population would be useful.

III. System Design

The data for this study were collected and recorded using CodeHelp software deployed in the course, two surveys, and class performance records. These surveys were provided to students at the start and end of the introductory programming course to assess their perception and attitude toward CodeHelp, learning, and the programming course. The surveys focused on students' demographics, perspectives on class structure, their motivation and preference, programming experience, their experience with CodeHelp, and its usage.

A. CodeHelp Deployment

CodeHelp was deployed only a few weeks following the start of the course. CodeHelp was accessible via a link on students' course homepages on the course's learning

management system, Canvas. By clicking this link, individual students' course profiles were automatically linked to their CodeHelp accounts. This automatic link allowed us to associate submitted queries and overall CodeHelp usage with the students' survey responses. The use of CodeHelp in this course was voluntary. It was advertised to students as a supplement to their education and a resource they could direct programming questions as they needed.

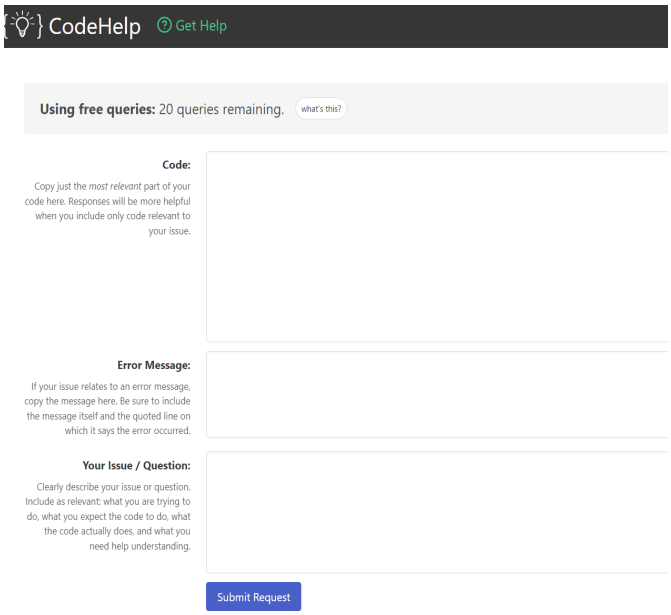


Fig. 1. A view of CodeHelp interface. Users can enter the code snippet they want to explore, the error message they get while executing the code, and the questions they might have regarding it.

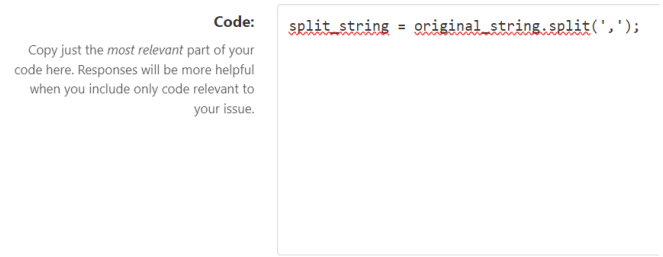


Fig. 2. An example of student code input in the text box with code prompt in the CodeHelp user interface.

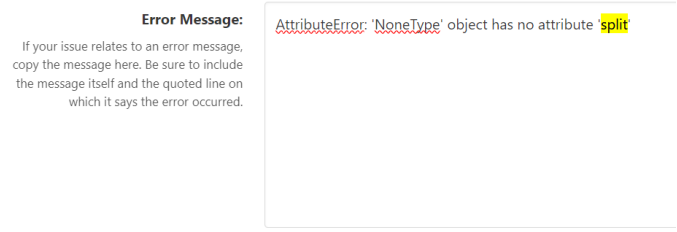


Fig. 3. An example of student error input in the text box with error message prompt in the CodeHelp user interface.

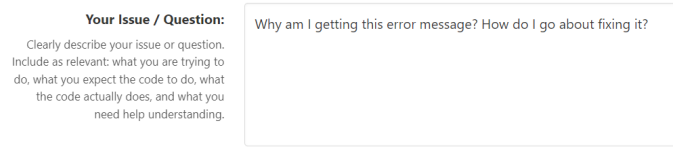


Fig. 4. An example of student issue input in the text box with the "your issue/question" prompt in the CodeHelp user interface.

The course instructor specified the programming languages available in the CodeHelp software, and students could select the language they were having issues with. Students could then input their desired questions into several text boxes. Fig. 1 illustrates the CodeHelp interface. The text box labeled "Code" allows users to input a specific section of code that they did not understand or that was giving them trouble. A sample of student input in the Code text box is illustrated in Fig. 2. The text box labeled "Error Message" allows users to paste or describe specific errors the student was running into, such as logic or syntax errors. Students would typically copy and paste error messages output by their Integrated Development Environment (IDE), particularly ones they did not comprehend or could not solve. A sample of student input in the Error Message text box is illustrated in Fig. 3. The "Your Issue/Question" text box allowed users to describe the issue they were having in as much detail as possible so that CodeHelp could properly analyze the problem and understand how to assist. A sample of student input in the Your Issue/Question text box is illustrated in Fig. 4.

B. Survey Data Collection

University's Institutional Review Board approval was obtained. To assess the effectiveness of CodeHelp and how supportive it is to students, two surveys were deployed through Google Forms in the course to capture students' initial perceptions (at the beginning of the course) and final reflections (at the end of the course). The survey questions were inspired by the computing attributes survey [36]. The first survey was conducted in the first week of the course, assigned as an assignment, and students had 4–5 days to complete it. This was a mandatory survey where students provided responses irrespective of whether they used the CodeHelp tool or not. The demographics collected in the survey established a baseline of students' background in the introductory programming course, how confident they are in programming, and their self-regulation and autonomy in their learning process. The second survey was required to be submitted along with the student's final project submission on Canvas at the very end of the semester. This survey served as a reflection on students' learning experiences, focusing on their usage of CodeHelp, their confidence in programming after this course, and their satisfaction with the course. It aimed to understand the usefulness of CodeHelp and the challenges associated with using it. Additionally, the open-ended questions in the second survey help collect qualitative feedback on why students chose to use or not use CodeHelp. All participants in this study were students in this introductory programming course, with a wide range of ages, experience levels, and backgrounds. Both surveys provided insights into the relationship between using AI-based programming assistance tools and students' perception and performance.

The data was stored in Excel spreadsheets for easy management and transferred to Statistical Package for the Social Sciences by International Business Machines Corporation (IBM SPSS), a Software for statistical analysis. SPSS was used to analyze quantitative data from both surveys. As the surveys are tied to their course grade, most students completed them. The high response rate allowed us to identify any significant changes in programming confidence and learning outcomes over time. In addition, the experience of students who used CodeHelp versus those who

did not was compared. This comparison explores how students leverage AI tools for learning programming and how it impacts their course performance.

C. Participants

A total of 508 students enrolled in the course participated in the study. The average age of participants in this study was 18.97 years, and the ages ranged from 17 to 31. Among the 508, a total of 361 students identified as 'male', 138 identified as 'female', and 9 students identified as other. Looking at the individual counts for each identity and race regardless of the combination, 257 students identified as White, 140 students identified as Asian, 119 students identified as Hispanic or Latino/Latina, 44 students identified as South Asian, and 30 students identified as Black African American. The number of students who aligned with other identities was 4 Middle Eastern, 2 Indian, 1 Arab, 1 Brazilian, 2 Native American, 1 Ukrainian, 1 West Indian (Trinidadian), 1 Native Hawaiian or Pacific Islander, 1 Ashkenazi/Sephardic, and 1 African.

The average Grade Point Average (GPA) for all students was 3.57 out of 4.0, and the average number of credits being taken per participant during this semester was 14.37 credits in a range from 4 to 20. The minimum number of credits a full-time student can take in one semester is 12, and the maximum is 18, with exceptions for students with disabilities or accommodations. Out of 508 students, 308 of these students, or 60.63%, secured an A as their final grade in the course. There were 2 exams in this course. The average exam scores for exam 1 were 81.94%, and the average scores for exam 2 were 88.47%, which is a 6.53% increase in overall performance. The 3 most common majors students in this introductory programming course had been Computer Science (50.39%), Computer Engineering (13.19%), and Data Science (7.87%). In this course, there were a majority of 337 freshmen, 98 sophomores, 53 juniors, and 12 seniors. The remaining students identified as 'other'.

IV. Data Analysis

All data from the course surveys and from CodeHelp were stored in Excel sheets after collection. Each student in the course was given a unique 4-digit ID to ensure student anonymity. Student survey responses and CodeHelp data were correlated using this ID, thereby linking student performance and CodeHelp usage. First the descriptive statistics for the usage of CodeHelp, its helpfulness, their prior experience, and the number of queries submitted is presented.

For analysis purposes, students were categorized into two groups: the group of students who used CodeHelp at least once are called 'users', and non-users were students who did not use CodeHelp even once. To assess the difference between users and non-users in terms of academic performance, an independent T-test on their Exam 2 scores (Exam 2 was administered at the end of the semester) was used. Exam scores were on a 100-point scale, with some students having grades over 100 due to 10 possible extra credit points. The final letter grades of users and non-users was also compared. The letter grades were converted to numerical values, where lower numbers indicate better letter grades (A = 2, A- = 3, B+ = 4, B = 5, B- = 6, C+ = 7, C = 8, C- = 9, D+ = 10, D = 11, D- = 12, F = 13).

A two-way Analysis of Variance (ANOVA) test was conducted to identify any interaction between exam time and CodeHelp use for exam scores. This will compare Exam 2 (end-term) scores with Exam 1 (mid-term) scores of users and non-users. Differences in attitude between CodeHelp users and non-users were assessed through a couple of survey items on a scale of 1 to 5. An independent T-test was run between the scores of CodeHelp users and non-users on questionnaire items. These items were: "I have good time management skills", "I set specific times for my study", "I am responsible for my own decision/actions", "I preferred to watch the lectures online and hardly attended lectures in-person", "Attending labs should be optional", "I often submitted my programs again after the auto grader reported some errors with some of the test cases", "I used CodeHelp multiple times to solve my programming doubts", "I found the CodeHelp responses helpful and accurate in answering my questions", and "I find programming to be challenging".

Further, to better understand who the students were using the CodeHelp tool, their demographics were investigated. A one-way ANOVA test was conducted within the population of all CodeHelp users to determine if demographic factors, such as ethnicity, gender, and age, and survey responses had any significant relationship with the number of queries submitted by users. Furthermore, changes in whether CodeHelp users' exam performance over the course of the semester were investigated. Further, Exam 1 and Exam 2 scores within CodeHelp users, along with the number of queries submitted was compared, in order to assess whether or not asking more queries in CodeHelp was correlated with better exam scores over time. The survey items were also used to assess usage and perception of CodeHelp. These items were as follows: "I used CodeHelp multiple times to solve my programming doubts" and "I found the CodeHelp responses helpful and accurate in answering my questions". The question "Did you use CodeHelp (the AI assistant) provided on canvas?".

V. Findings and Discussion

Among 508 students enrolled in the course, only 30% of the students (155 students) used CodeHelp by creating an account and submitting at least one query. This shows that not the majority of students felt self-motivated to explore and use the AI-based coding assistant tool. Possibly, many students felt confident to improve performance with the usual course material, and some lacked motivation to put in work to learn programming outside class. In terms of programming experience, out of 508 students, 30% reported having no prior experience in coding before taking this class, 20% had between 1 to 10 h of programming experience, 39% had between 11 to 100 h, 9% had 101 to 500 h, and 1% identified as software developers. This data provides valuable information for understanding the distribution of study participants based on prior programming experience. This distribution with many students having lower prior programming experience is not surprising, as this was an introductory programming course.

On average, each CodeHelp user (students who used CodeHelp) submitted 11.6 queries. In the second survey, participants responded to "I used CodeHelp multiple times to

solve my programming doubts." This item received an average rating of 2.65 out of 5.0. Further, they were asked to rate their experience with CodeHelp using "I found the CodeHelp responses helpful and accurate in answering my questions" on a scale of 1 to 5. This item received a slightly higher average of 2.75 out of 5.0. The slightly above the neutral point of the rating scale shows that students engaged with the tool to address their questions and found it somewhat helpful and accurate when seeking support in their programming endeavors. When asked about their feedback on an open-ended question about users' experience using CodeHelp, students provided a mixed response, yet the majority had a positive outlook on CodeHelp's utility. In their open-ended responses about their experience of using CodeHelp, several students appreciated CodeHelp for its ability to break down their programming errors. One participant mentioned, "CodeHelp was fantastic... It pointed out the flaws detected. walks through the logic of what the user is attempting to do, and gives ideas to the user about how it could be solved". This response illustrates that CodeHelp is perceived as helpful by some students through constructive feedback that fosters deeper understanding rather than simply offering direct solutions. This also shows how such a tool can inspire reflection and better engagement with the problem. Some students believed CodeHelp was reliable for their learning process. One student shared, "I used it frequently for clarification. It helped me understand why my logic was wrong... it can quickly provide us with the necessary feedback to fix our code." While many of the students provided positive feedback about CodeHelp, some found it less helpful, potentially due to the restrictive nature of CodeHelp, which does not provide direct solutions and provides hints and feedback to students to solve the problem themselves. Some of them possibly found that the responses that CodeHelp provided were too broad. Overall, students agreed that CodeHelp was especially useful when no other resources were available, with one student stating, "CodeHelp was a great resource for when I was stuck, and there were no office hours".

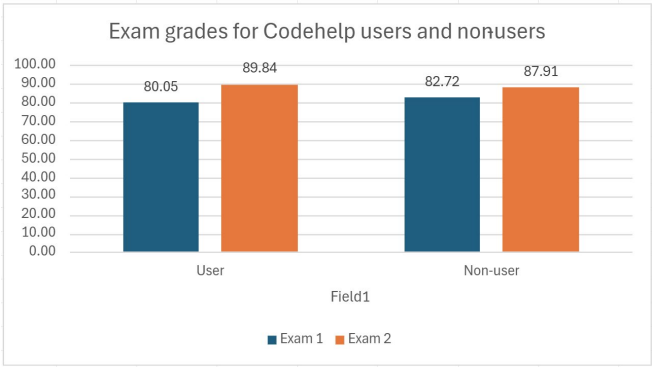


Fig. 5. The average exam grade (out of 100) between CodeHelp users and non-users.

A. Differences between CodeHelp Users and Non-users

Significant differences were observed within the T-tests conducted on various factors are discussed. Although no significant difference was observed between users' and non-users' exam grades, there was a slightly greater improvement in exam performance from Exam 1 to Exam 2 for users compared to non-users. CodeHelp users scored a mean =80.05 in Exam 1 and scored a mean of 89.84 in Exam 2, whereas non-users scored a mean of 82.72 in Exam 1 and a mean of 87.91 in Exam 2 as shown in Fig. 5. The two-way ANOVA showed an almost significant interaction between exam time (Exam 1 or Exam 2) and using CodeHelp (user or non-user) on exam scores F (1, 1010) = 3.60, p = 0.058, shown in Fig. 6. Additionally, final course grade data demonstrated that CodeHelp users (M = 2.77, SD = 1.48) achieved higher grades in the course than non-users (M = 3.28, SD = 2.29), t(505) = 2.54, p = 0.011 (also reflected in Fig. 7). A Cohen's d = 0.24 was observed for the test, showing a small effect size. Here, a lower mean value represents stronger academic performance, where A corresponds to 2, A- corresponds to 3, and B+ corresponds to 4. Fig. 7 demonstrates that the final grades of CodeHelp users appear to be higher than those of non-users. In this scenario, an average final grade of 2.77 for CodeHelp users would translate to be approximately between an A (or 2) or A- (or 3), while the average final grade of 3.28 for nonusers would translate approximately between an A- (or 3) and B+ (or 4). This indicates that engaging with CodeHelp may have positively influenced students' overall academic success. These results may also suggest that students who are academically inclined or work harder to achieve higher performance, are the ones using CodeHelp. In their pursuit of greater learning, they explored and used CodeHelp.

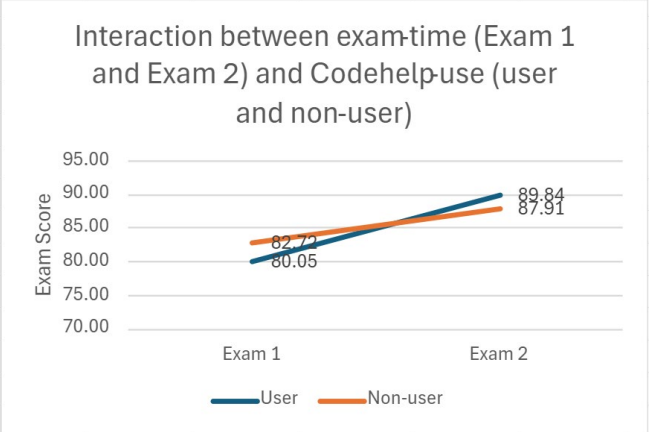


Fig. 6. The Exam 1 and Exam 2 grade between CodeHelp users and non-users.

The relationship between CodeHelp usage and students' academic backgrounds and self-management skills was examined. CodeHelp users reported stronger time management skills than non-users even before the course began, as indicated by the survey conducted at the beginning of the semester. This means that students with better time management skills tend to explore and use CodeHelp. This aligns with the expectation that self-regulated help-seeking behavior is related to individuals' motivation [16]. Better time management may have helped them explore and use CodeHelp more. Individuals who demonstrate better time management skills tend to be more self-regulated learners and, therefore, are more likely to use the additional AI tool to learn programming. Therefore, the observation aligns with self-instruction, time management, and strategies of self-regulated learning [16]. The results of the T-test showed users averaging 3.84 (SD = 0.83) for users versus 3.62 (SD = 0.98) for non-users, t(500) = −2.46, p = 0.014. This

was observed with a small effect size, Cohen's $d = -0.24$. CodeHelp users also tended to set specific study times more consistently for users ($M = 3.52$, $SD = 0.99$) compared to non-users ($M = 3.28$, $SD = 1.13$), $t(500) = -2.25$, $p = 0.025$. A Cohen's $d = -0.22$ was observed for the test, showing a small effect size. Furthermore, users reported a higher sense of responsibility and greater accountability for their actions ($M = 4.70$, $SD = 0.46$) than non-users ($M = 4.56$, $SD = 0.67$), $t(500) = -2.32$, $p = 0.021$, and Cohen's $d = -0.22$. In addition, CodeHelp users generally had higher GPAs ($M = 3.65$, $SD = 0.46$) than their non-using peers ($M = 3.54$, $SD = 0.53$), $t(474) = -2.35$, $p = 0.019$, Cohen's $d = -0.23$, suggesting that students with stronger academic backgrounds were more likely to adopt CodeHelp as a tool to support their learning. This aligns with prior research indicating that self-regulated learning tools can support students in exploring assignments and seeking answers to their questions at their own pace [29].

In terms of engagement with learning resources, CodeHelp users ($M = 4.88$, $SD = 0.35$) attended weekly labs more consistently than non-users ($M = 4.70$, $SD = 0.69$), $t(302) = -2.45$, $p = 0.015$, Cohen's $d = -0.29$. Non-users ($M = 3.31$, $SD = 1.47$) were also more likely to feel that lab attendance should be optional than CodeHelp users ($M = 2.76$, $SD = 1.31$), $t(302) = 3.26$, $p = 0.001$, Cohen's $d = 0.39$. Users showed less reliance on online lectures for flexibility ($M = 3.55$, $SD = 1.43$) compared to non-users ($M = 4.08$, $SD = 1.23$), $t(302) = 3.31$, $p < 0.001$, Cohen's $d = 0.39$, indicating that non-users may prefer a more flexible learning approach, whereas CodeHelp users prefer to benefit from combining online and in-person resources.

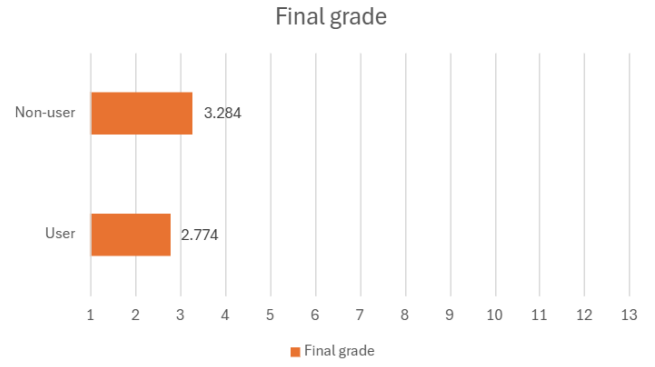


Fig. 7. The average final course grade between CodeHelp users and non-users. Reference for the conversion of letter grades to numbers (Letter grade to number conversion A = 2, A- = 3, B+ = 4, B = 5, B- = 6, C+ = 7, C = 8, C- = 9, D+ = 10, D = 11, D- = 12, F = 13).

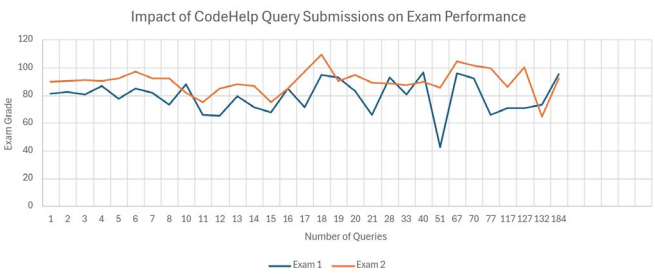


Fig. 8. Impact of CodeHelp on students' midterm (exam 1) and end-term (exam 2) exam scores.

CodeHelp users also perceived the tool as more helpful and accurate ($M = 2.75$, $SD = 2.09$) compared to non-users ($M = 0.39$, $SD = 1.21$), $t(302) = -12.42$, $p < 0.001$, Cohen's $d = -1.48$. This was expected as students who considered CodeHelp would only use it to learn programming. Students who perceived it as neither helpful nor accurate did not use it. These findings collectively suggest that CodeHelp users not only engage more actively with their coursework but also see the tool as a valuable resource for supporting their learning and progress in programming. In line with prior investigation of CodeHelp [14], the study also found a generally positive association between the use of CodeHelp and overall performance and self-management.

B. Differences within CodeHelp Users

Within CodeHelp users, 76% of them (118 out of 155) reported using CodeHelp multiple times. This may be an indication that if students are exposed to such a programming assistance tool and try it out, many of them would use it multiple times. As shown in the graph in Fig. 8, the X-axis represents the 'Number of Queries Submitted by Students through CodeHelp,' and the Y-axis shows the 'Average Exam Scores for CodeHelp Users'. Students began utilizing CodeHelp before Exam 1, but their use of CodeHelp increased significantly by Exam 2. This increased usage correlates with higher rise scores on Exam 2 from Exam 1. Furthermore, the rise in the number of submitted queries appears to somewhat align with higher scores, particularly evident in Exam 2. This trend suggests that greater engagement with CodeHelp positively impacts student performance, as submitting more queries likely provides students with clarity and support on challenging topics, contributing to improved exam outcomes. In the graph, a dip can be observed at the end. However, this can be ignored as an anomaly.

The one-way ANOVA conducted within all CodeHelp users did not report any significant differences in academic performance or classroom experience depending on the level of CodeHelp usage. Factors such as exam scores, final grades, GPA, ethnicity, gender, and age did not correlate with the level of CodeHelp usage (single use vs multiple use) among CodeHelp users themselves. Questions regarding self-management skills and classroom participation were also not found to be significantly different among CodeHelp users. In the end-of-semester survey questionnaire, a significant difference was found between single-time CodeHelp users and multiple-time CodeHelp users on questions asking if they used CodeHelp multiple times ($p < 0.001$), and finding CodeHelp useful ($p = 0.002$), which is expected as these were the students who used CodeHelp multiple times and found it helpful. Or the students who find it useful did use it multiple times. This observation aligns with prior research indicating that help-seeking behaviors in the use of AI systems are shaped by factors such as perceived usefulness [37]. Recently, there has been a substantial exploration of ChatGPT as a generative AI tool to support education [16]. Prior work has demonstrated the positive potential of using AI, such as Generative AI in education to support student engagement, learning experience, learning outcomes, collaborative work, and so on [38–41]. Our work also demonstrates the potential use of AI and shows evidence that generative AI may support learning programming and

may support learning performance in introductory programming courses. Overall, the study results supports the use of AI-based programming assistant tools to improve academic performance, or students inclined to work towards improving their academic performance resort to such tools and find them useful. Overall, the exploration found the response to our research question RQ1. Our results indicate that possibly generative AI-based tools for assisting in learning programming could be an instrumental tool in improving students' performance and learning, as observed through the better scores from Exam 1 to Exam 2. Our exploration also found the response to the research question RQ2. The results show that users of CodeHelp are the people who take the initiative and are proactive; they are the ones who attend weekly labs consistently and are the ones who are more comfortable with the attendance collection in weekly labs. CodeHelp users were the ones who showed a preference for in-person classes and a lesser inclination for online and hybrid setups. It was also observed that CodeHelp users performed somewhat better on academic assessment and showed greater improvement compared to non-users. It indicates that CodeHelp users are those who strive towards better performance and choose to explore support tools to improve their skills. CodeHelp users are also the ones who are good at managing their time. Our results are summarized in Table 1. Based on the study findings, it can be said that generative AI-based tools have substantial potential to augment learning and provide aid to those wanting to improve their learning and academic performance [40]. Evidence of generative AI as a tool to support formal learning has been discussed in the literature [42]. While there are evident benefits of using Generative AI-based to support learning, it is definitely not a replacement for the instructor. It's rather an additional support tool that can help with learning and potentially make learning more personalized.

Table 1. Summary of results of differences between CodeHelp users vs. non-users

Results	p-value	Mean and SD CodeHelp users	Mean and SD non-users
CodeHelp users secured significantly higher grades than non-users	p = 0.011	M=2.77, SD=1.48	M=3.28, SD=2.29
CodeHelp users reported stronger time management skills than non-users	p = 0.014	M=3.84, SD=0.83	M=3.62, SD=0.98
CodeHelp users reported setting study times more consistently than non-users	p = 0.025	M=2.77, SD=1.48	M=3.28, SD=2.29
Codehelp users reported higher sense of responsibility than non-users	p = 0.021	M=4.70, SD=0.46	M=4.56, SD=0.67
CodeHelp users generally had higher GPA than non-users	p = 0.01	M=3.65, SD=0.46	M=3.54, SD=0.53
Almost significant interaction between exam time and using CodeHelp	p = 0.058		

VI. Conclusion and Limitations

This study examined the use of CodeHelp, a Generative AI-based coding assistant, in an introductory programming course. Our work also demonstrates the potential use of AI and shows evidence that generative AI may support learning programming and may support learning performance in introductory programming courses. Overall, it can be said that the use of AI-based programming assistant tools can provide scaffolding to students inclined to work towards improving their programming skills and academic performance. Further, they also tend to find such tool's support useful and therefore use it multiple times.

In response to research question RQ1, the results indicate that guardrailed generative AI-based tools for assisting in learning programming can assist early programmers. The tool provided a platform for students to practice programming and assist in solving their programming errors on their own. Results show that CodeHelp users showed greater improvement from Exam 1 to Exam 2 compared to non-users, evident from Fig. 5. CodeHelp users also had higher course grades compared to non-users.

In response to research question RQ2, the results show that users of CodeHelp are the students who take the initiative to learn and are proactive; they are the ones who attend weekly labs consistently and are the ones who are more comfortable with the attendance collection in weekly labs. CodeHelp users were the ones who showed a preference for in-person classes and a lesser inclination for online and hybrid setups. This is another indication that students who put more effort into learning by showing up in class and labs are also the ones willing to explore and use such tools. Findings also suggest that CodeHelp users are willing to work harder and improve their performance and skills, in alignment with self-regulated learning driven by interest and self-control. These results are particularly valuable as the exploration was conducted in real classroom settings with large class enrollment. This is especially helpful where one-to-one interaction with the instructor is limited. CodeHelp users were found to be more self-reliant in a classroom setting. They typically had better time management skills and perceived themselves as responsible.

The benefits of such AI assistance seem to depend on individuals in a voluntary setting. Potentially, there may be a group of students who, if provided more exposure to technology and gather awareness, would also reap the benefits of such tools to learn better. This indicates a potential area of improvement. Therefore, future work in this direction should investigate in what ways students benefit from using such tools. The results of the study provide further motivation for researchers and developers to use LLMs and develop even better tools that provide students with support to learn programming in introductory computer science courses and possibly extend such tools for non-computer science students to learn programming. Such assistance could be particularly useful to novice programmers who often run into problems frequently during programming and need more personalized support.

There are some limitations of the study. Conducting the study in a natural classroom environment restricted control over the experimental design, including the ability to ensure equivalent group sizes between CodeHelp users and non-users. Additionally, CodeHelp usage was voluntary, which naturally creates unequal groups. However, this is not detrimental, particularly given that research situated in authentic learning contexts often yields insights with stronger ecological validity than those obtained under controlled conditions. We limited our investigation to self-reported data due to constraints imposed by classroom-based research. The study design enabled the identification of the characteristics

of students who choose to use such tools, aligning directly with the study's primary objectives.

CONFLICT OF INTEREST

The authors declare no conflict of interest

AUTHOR CONTRIBUTIONS

Ashish Aggarwal ideated the project, brainstormed study details, and collected the data; Neha Rani, Anh Le, and Joanna Mijares conducted the qualitative and quantitative data analysis and drafted the manuscript. All authors have reviewed and approved the manuscript.

FUNDING

This work was supported by the University of Florida Presidential Strategic Initiative Grant.

REFERENCES

[1] S. Arif, S. A. Kashani, M. Mukhtyar *et al*., "Trends in higher education enrollment in computer sciences (2012-2017) and emerging job opportunities," *Contemporary Journal of Social Science Review*, vol. 2, no. 4, pp. 1691–1700, 2024.

[2] K. E. Boyer, R. S. Dwight, C. S. Miller, C. D. Raubenheimer, M. F. Stallmann, and M. A. Vouk, "A case for smaller class size with integrated lab for introductory computer science," *ACM SIGCSE Bulletin*, vol. 39, no. 1, pp. 341–345, 2007. doi: 10.1145/1227504.1227430

[3] C. Mulryan-Kyne, "Teaching large classes at college and university level: Challenges and opportunities," *Teaching in Higher Education*, vol. 15, no. 2, pp. 175–185, 2010. doi: 10.1080/13562511003620001

[4] P. C. Fortes and A. Tchantchane, "Dealing with large classes: A real challenge," *Procedia-Social and Behavioral Sciences*, vol. 8, pp. 272–280, 2010. doi: 10.1016/j.sbspro.2010.12.037

[5] A. Ghimire and J. Edwards, "Coding with AI: How are tools like ChatGPT being used by students in foundational programming courses," in *Proc. International Conference on Artificial Intelligence in Education*. Springer, 2024, pp. 259–267. doi: 10.1007/978-3-031-64299-9_20

[6] P. Talwar, *The Strategic Use of Virtual Teacher Assistants to Increase Faculty Engagement and Job Satisfaction*, Trident University International, 2023.

[7] A. Alam, "Should robots replace teachers? Mobilisation of AI and learning analytics in education," in *Proc.* 2021 *International Conference on Advances in Computing, Communication, and Control* (ICAC3), IEEE, 2021, pp. 1–12. doi: 10.1109/ICAC353642.2021.9697300

[8] E. Kasneci, K. Seßler, S. Kuchemann *et al*., "ChatGPT for good? On opportunities and challenges of large language models for education," *Learning and individual differences*, vol. 103, 102274, 2023. doi: 10.1016/j.lindif.2023.102274

[9] N. Rani, Y. Qian, and S. L. Chu, "Explanation for user trust in context-aware recommender systems for search-as-learning," in *Proc. 2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, IEEE, 2023, pp. 47–49. doi: 10.1109/ICALT58122.2023.00019

[10] N. Rani, S. L. Chu, and V. R. Mei, "Investigating the effects of different levels of user control on the effectiveness of context-aware recommender systems for web-based search," in *Proc. CHI Conference on Human Factors in Computing Systems Extended Abstracts*, 2022, pp. 1–6. doi: 10.1145/3491101.3519802

[11] G. Pitts, N. Rani, W. Mildort, and E.-M. Cook, "Students' reliance on ai in higher education: identifying contributing factors," in *Proc. International Conference on Human-Computer Interaction*, Cham: Springer Nature Switzerland, 2025, pp. 86–97. doi: 10.1007/978-3-032-12773-0_9

[12] N. Rani, E. Cenko, and L. M. C. Castro, "Perceived Importance of Cognitive Skills Among Computing Students in the Era of AI," arXiv preprint, arXiv:2604.10730, 2026.

[13] M. Liffiton, B. E. Sheese, J. Savelka, and P. Denny, "Codehelp: Using large language models with guardrails for scalable support in programming classes," in *Proc. the 23rd Koli Calling International Conference on Computing Education Research*, 2023, pp. 1–11. doi: 10.1145/3631802.3631830

[14] B. Sheese, M. Liffiton, J. Savelka, and P. Denny, "Patterns of student help-seeking when using a large language model-powered programming assistant," in *Proc. the 26th Australasian Computing Education Conference*, 2024, pp. 49–57. doi: 10.1145/3636243.3636249

[15] P. Denny, J. Prather, B. A. Becker, J. Finnie-Ansley, A. Hellas, J. Leinonen, A. Luxton-Reilly, B. N. Reeves, E. A. Santos, and S. Sarsa, "Computing education in the era of generative AI," *Communications of the ACM*, vol. 67, no. 2, pp. 56–67, 2024. doi: 10.1145/3624720

[16] T. K. Chiu, "A classification tool to foster self-regulated learning with generative artificial intelligence by applying self-determination theory: A case of ChatGPT," *Educational Technology Research and Development*, pp. 1–16, 2024. doi: 10.1007/s11423-024-10366-w

[17] S. Lau and P. Guo, "From "ban it till we understand it" to "resistance is futile": How university programming instructors plan to adapt as more students use ai code generation and explanation tools such as ChatGPT and github copilot," in *Proc. the 2023 ACM Conference on International Computing Education Research,* vol. 1, 2023, pp. 106–121. doi: 10.1145/3568813.3600138

[18] J. O. McGinnis, "Accelerating ai," *Nw. L. Rev. Colloquy*, vol. 104, 366, 2009.

[19] N. Rani, S. Majumder, I. Bhardwaj, and P. G. F. Garcia. "Can AI support student engagement in classroom activities in higher education?" arXiv Preprint, arXiv:2506.18941, 2025.

[20] A. Tamkin, M. Brundage, J. Clark, and D. Ganguli, "Understanding the capabilities, limitations, and societal impact of large language models," *CoRR*, vol. abs/2102.02503, 2021.

[21] M. J. Parker, C. Anderson, C. Stone, and Y. Oh, "A large language model approach to educational survey feedback analysis," *International Journal of Artificial Intelligence in Education*, pp. 1–38, 2024. doi: 10.1007/s40593-024-00414-0

[22] I. Estevez-Ayres, P. Callejo, M. A. Hombrados-Herrera, C. Alario-Hoyos, and C. Delgado Kloos, "Evaluation of llm tools for feedback generation in a course on concurrent programming," *International Journal of Artificial Intelligence in Education*, pp. 1–17, 2024.

[23] N. V. Covington and O. Vruwink, "ChatGPT in undergraduate education: Performance of gpt-3.5 and identification of ai-generated text in introductory neuroscience," *International Journal of Artificial Intelligence in Education,* pp. 1–24, 2024. doi: 10.1007/s40593-024-00427-9

[24] F. Zhang, C. Li, O. Henkel, W. Xing, S. Baral, N. Heffernan, and H. Li, "Math-llms: Ai cyberinfrastructure with pre-trained transformers for math education," *International Journal of Artificial Intelligence in Education*, pp. 1–24, 2024. doi: 10.1007/s40593-024-00416-y

[25] C. Erta and N. Rani, "Reframing Aging Through AI-Generated Patient Simulations in Pre-Medical Education," *Innovation in Aging*, vol. 9, no. Supplement_2, 2025. doi: 10.1093/geroni/igaf122.409

[26] N. Rani, J. R. Munnangi, A. M. Spangler, and D. Honeycutt, "Engineering Students' Usage and Perceptions of GitHub Copilot in Open-Source Projects," arXiv Preprint, arXiv:2604.10747, 2026.

[27] D. Hennekeuser, D. D. Vaziri, D. Golchinfar, D. Schreiber, and G. Stevens, "Enlarged education–exploring the use of generative ai to support lecturing in higher education," *International Journal of Artificial Intelligence in Education*, pp. 1–33, 2024. doi: 10.1007/s40593-024-00424-y

[28] J. Leinonen, A. Hellas, S. Sarsa, B. Reeves, P. Denny, J. Prather, and B. A. Becker, "Using large language models to enhance programming error messages," in *Proc. the 54th ACM Technical Symposium on Computer Science Education,* vol. 1, 2023, pp. 563–569. doi: 10.1145/3545945.3569770

[29] R. Yilmaz and F. G. K. Yilmaz, "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*, vol. 4, 100147, 2023. doi: 10.1016/j.caeai.2023.100147

[30] A. Shah, A. Chernova, E. Tomson, L. Porter, W. G. Griswold, and A. G. S. Raj, "Students' use of github copilot for working with large code bases," in *Proc. the 56th ACM Technical Symposium on Computer Science Education,* vol. 1, 2025, pp. 1050–1056. doi: 10.1145/3641554.3701800

[31] R. Ferdiana, "The impact of artificial intelligence on programmer productivity," in *Proc. International Conference on Software Engineering and Information Technology (ICOSEIT)*, February 2024.

[32] J.-M. Su and F.-Y. Hsu, "Building a visualized learning tool to facilitate the concept learning of object-oriented programming," in *Proc. 6th IIAI International Congress on Advanced Applied Informatics* (IIAI-AAI), 2017, pp. 516–520. doi: 10.1109/IIAI-AAI.2017.180

[33] M. Kazemitabaar, R. Ye, X. Wang, A. Z. Henley, P. Denny, M. Craig, and T. Grossman, "Codeaid: Evaluating a classroom deployment of an llm-based programming assistant that balances student and educator needs," in *Proc. the 2024 Chi Conference on Human Factors in Computing Systems*, 2024, pp. 1–20. doi: 10.1145/3613904.3642773

[34] S. MacNeil, J. Kim, J. Leinonen *et al*., "The implications of large language models for cs teachers and students," *SIGCSE,* 1255, 2023. doi: 10.1145/3545947.3573358

[35] B. Sheese, M. Liffiton, J. Savelka, and P. Denny, "Patterns of student help-seeking when using a large language model-powered programming assistant," in *Proc. the 26th Australasian Computing Education Conference*, pp. 49–57. 2024. doi: 10.1145/3636243.363624

[36] B. Dorn and A. E. Tew, "Empirical validation and application of the computing attitudes survey," *Computer Science Education*, vol. 25, no. 1, pp. 1–36, 2015. doi: 10.1080/08993408.2015.1014142

[37] D. Adams, K. M. Chuah, E. Devadason, and M. S. A. Azzis, "From novice to navigator: Students' academic help-seeking behaviour, readiness, and perceived usefulness of ChatGPT in learning," *Education and Information Technologies*, vol. 29, no. 11, pp. 13617–13634, 2024. doi: 10.1007/s10639-023-12427-8

[38] C. K. Lo, K. F. Hew, and M. S.-Y. Jong, "The influence of ChatGPT on student engagement: A systematic review and future research agenda," *Computers & Education*, 105100, 2024. doi: 10.1016/j.compedu.2024.105100

[39] O. A. Khurma, F. Albahti, N. Ali, and A. Bustanji, "Ai ChatGPT and student engagement: Unraveling dimensions through prisma analysis for enhanced learning experiences," *Contemporary Educational Technology*, vol. 16, no. 2, ep503, 2024. doi: 10.30935/cedtech/14334

[40] Y. M. E. HEUNG and T. K. Chiu, "How ChatGPT impacts student engagement from a systematic review and meta-analysis study," *Computers and Education: Artificial Intelligence*, 100361, 2025. doi: /10.1016/j.caeai.2025.100361

[41] G. Huesca, Y. Martınez-Trevino, J. M. Molina-Espinosa, A. R. Sanroman-Calleros, R. Martınez-Roman, E. A. Cendejas-Castro, and R. Bustos, "Effectiveness of using ChatGPT as a tool to strengthen benefits of the flipped learning strategy," *Education Sciences*, vol. 14, no. 6, 660, 2024. doi: 10.3390/educsci14060660

[42] B. K. Younis, "Examining students' self-regulation skills, confidence to learn online, and perception of satisfaction and usefulness of online classes in three suggested online learning environments that integrates ChatGPT." *Online Learning*, vol. 28, no. 2, 2024. doi: 10.24059/olj.v28i2.4397

Copyright © 2026 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (CC BY 4.0).