Creative Education, 2026, 17(1), 76-101
https://www.scirp.org/journal/ce
ISSN Online: 2151-4771
ISSN Print: 2151-4755

The Impact of Generative Artificial Intelligence on College Students' Computer Thinking in the Task of Complex Computer Programming: Based on Social Cognitive Theory

Wenhui Yin

Department of Information Engineering, Inner Mongolia Mechanical and Electrical Vocational Technical College, Hohhot, China
Email: ywh2016@126.com

How to cite this paper: Yin, W. H. (2026). The Impact of Generative Artificial Intelligence on College Students' Computer Thinking in the Task of Complex Computer Programming: Based on Social Cognitive Theory. *Creative Education, 17,* 76-101.
https://doi.org/10.4236/ce.2026.171007

Received: October 11, 2025
Accepted: January 19, 2026
Published: January 22, 2026

Abstract

Generative Artificial Intelligence (GAI) is rapidly reshaping programming education, yet little is known about how college students cognitively, emotionally, and behaviorally engage with AI during complex programming tasks. Grounded in Social Cognitive Theory, this qualitative study explores how the use of GAI influences computational thinking (CT) among 16 undergraduates enrolled in a 15-week programming course supported by AI tools. Through semi-structured interviews and thematic analysis, five core themes emerged: (1) Cognitive Processing, in which AI scaffolds task decomposition, logical reasoning, multi-solution comparison, and iterative debugging while also introducing risks of cognitive substitution; (2) Emotional Experience, characterized by fluctuating self-efficacy, reduced frustration, heightened motivation, and anxiety triggered by AI errors or unpredictability; (3) Behavioral Strategies, where students adopt hybrid human-AI problem-solving pathways involving independent attempts, structured prompting, verification loops, and comparison-based reasoning; (4) Human-AI Collaboration Models, reflecting dynamic role negotiation in which students act as planners and evaluators while AI functions as solver, debugger, and explainer; and (5) Limitations and Risks, including error accumulation, misleading explanations, lack of contextual awareness, and emerging dependency concerns. These findings demonstrate that GAI operates as a cognitive partner that reshapes students' CT development, self-regulatory behaviors, and learning identities. The study extends Social Cognitive Theory into human-AI interaction contexts by illustrating triadic reciprocity among personal factors, environmental AI feedback, and adaptive behavioral strategies.

Keywords

Generative Artificial Intelligence (GAI), Computational Thinking (CT), Social Cognitive Theory, Complex Computer Programming

1. Introduction

There are currently initiatives in computer programming education to investigate how to improve teaching methods (Garcia, 2025). These teaching methods are significant for the development of human beings. People who are proficient in computer programming, an essential talent for many business sectors (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b), can develop new technologies that stimulate economic expansion and innovation (Eteng et al., 2022). According to the researcher's experiences, computer programming is the best course for fostering students' computational thinking (CT). This view aligns with decades of scholarship recognizing programming as a central pathway for cultivating CT competencies. Over the past 20 years, CT—first conceptualized by Papert (1980)—has garnered significant attention and in-depth discussion (Chen et al., 2024), not only because it enhances learners' coding abilities but also because it has evolved into a transferable method of problem-solving that extends far beyond computer science (Yadav et al., 2016). In other words, CT equips students with a systematic way of analyzing problems, designing solutions, and applying logical reasoning—skills that contribute directly to academic success, workplace adaptability, and long-term professional development. If students in colleges master this method of problem-solving, they will be more capable of navigating complex challenges throughout their lives.

In recent years, the rapid rise of Generative Artificial Intelligence (GAI) has fundamentally reshaped the landscape of programming education, creating new opportunities for cultivating CT. Unlike traditional tools that simply support coding tasks, GAI—particularly large language models such as ChatGPT—actively participates in the reasoning, debugging, and solution-generation process. This shift suggests that CT development may increasingly depend on how students interact with AI systems, how they make sense of AI-generated feedback, and how they integrate AI outputs into their own problem-solving strategies. Therefore, researching the impact of Generative Artificial Intelligence on college students' computational thinking is not only timely but also necessary for understanding the future of programming education. As AI becomes embedded in everyday learning environments, investigating its role in shaping CT provides both an important academic contribution and a practical opportunity to redesign how complex computer programming is taught.

Recent research has increasingly explored the educational affordances of large language model (LLM)-based generative AI tools, particularly their potential to enhance computational thinking (CT), programming competency, and language learning. A growing body of experimental, qualitative, and design-based studies

reveals a consistent pattern: generative AI, especially ChatGPT, can significantly transform how learners think, problem-solve, and interact with computational tasks. Across multiple empirical studies, CT emerges as the core theoretical anchor. Experimental findings demonstrate that integrating ChatGPT into programming courses substantially enhances students' computational thinking abilities, including creativity, algorithmic reasoning, problem-solving, and collaboration (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b; Yan et al., 2025; Gong et al., 2025). Notably, Gong et al. (2025) identify algorithmic thinking as the foundational CT skill strengthened by generative progressive prompts. Complementing these results, Yan et al. (2025) show that incorporating LLMs into collaborative programming not only improves CT but also reduces students' cognitive load, suggesting that AI support allows learners to allocate more cognitive resources to higher-order reasoning. Similar frameworks emphasizing human–AI interaction—such as Zhao et al.'s (2025) dialogue-negotiated programming model—highlight how structured prompt design and multi-agent interaction help cultivate deeper computational problem-solving behaviors.

Beyond CT, generative AI appears to influence important psychological and affective dimensions of learning. Yilmaz and Karaoglan Yilmaz (2023a, 2023b) report that ChatGPT-supported programming instruction significantly increases students' programming self-efficacy and motivation, suggesting that AI-mediated scaffolding can create more engaging and confidence-building learning environments. Their qualitative findings further indicate that learners acknowledge both benefits and risks of ChatGPT use, emphasizing the need for guided implementation. Extending AI's impact beyond programming domains, Liu et al. (2024) show that generative AI affects the composition processes of EFL learners, reinforcing the broader applicability of AI-mediated support in language-based learning tasks.

In addition to empirical studies, conceptual and design-based scholarship advances new frameworks for integrating LLMs into education. Hsu (2025) proposes a constructionist prompting framework that aligns CT elements with principles of prompt engineering, enabling learners to iteratively refine prompts and engage in naturalistic computational reasoning. Similarly, Zhao et al. (2025) advocate for human-computer collaborative programming through dialogue-negotiated problem solving, positioning generative AI as an active cognitive partner rather than a passive tool.

Although existing studies consistently show that generative AI can enhance computational thinking, programming performance, and learning motivation, a closer cross-study comparison reveals several structural limitations that constrain the robustness, generalizability, and theoretical development of current findings. These limitations emerge across three interconnected dimensions: research design, sample and context diversity, and longitudinal explanatory power.

First, methodological limitations are a pervasive issue across the literature. Experimental studies such as Yilmaz & Karaoglan Yilmaz (2023a, 2023b) and Yan et al. (2025) employ short intervention periods—five and four weeks respectively—

which restrict the ability to detect delayed cognitive, motivational, or behavioral effects of AI-supported programming. While these studies demonstrate short-term gains in CT or temporary fluctuations in self-efficacy, they leave open whether such effects stabilize, intensify, or reverse over time. Similarly, Gong et al. (2025) and Kohen-Vacs et al. (2025) focus primarily on immediate learning behaviors, offering little insight into sustained development of complex competencies such as deep algorithmic reasoning, adaptive problem-solving, or professional readiness in AI-rich learning environments. In parallel, Liu et al. (2024) rely solely on qualitative protocols without instrument validation, limiting the scalability and reliability of their findings. Collectively, these methodological gaps prevent the field from establishing causal or longitudinal claims about the educational impact of generative AI.

Second, across the reviewed literature, the impact of AI on college students' computational thinking in the task of complex computer programming remain insufficiently examined. In this article, complex programs in computer refer to systematic projects comprising numerous interrelated and interdependent subtasks, whose resolution requires students to possess systematic thinking and complex computational thinking capabilities. Only short-term intervention effects are documented, with little understanding of how learners' CT skills, self-efficacy, collaborative behaviors, or AI-mediated problem-solving strategies evolve with sustained exposure to generative AI tools. For example, while Yan et al. (2025) observe a temporary decline in self-efficacy, it is unclear whether this reflects an initial adjustment phase, a stable negative impact, or a transitional stage preceding deeper mastery. Similarly, studies rarely investigate how students gradually internalize prompt-design strategies, negotiation patterns with AI systems, or metacognitive adjustments—mechanisms that are conceptually important for understanding human–AI co-learning models but remain empirically unexamined. Even systematic reviews such as Massaty et al. (2024) highlight these gaps yet lack empirical follow-up.

Given the methodological and theoretical gaps identified in prior research, the present study aims to provide a deeper and more contextualized understanding of how generative artificial intelligence shapes college students' computational thinking in the task of complex computer programming. While existing studies primarily rely on short-term experiments, small samples, or surface-level behavioral measures, little is known about the underlying cognitive, emotional, and social mechanisms through which students interact with AI and gradually construct CT competencies. To address this limitation, this research adopts in-depth interviews as the core method, allowing for a rich exploration of students' lived experiences, reasoning processes, and self-regulatory strategies when engaging with generative AI tools. Grounded in Social Cognitive Theory, the study investigates how personal factors (e.g., self-efficacy, metacognitive monitoring), environmental influences (e.g., AI feedback, task complexity), and behavioral patterns (e.g., debugging with AI, prompt negotiation) dynamically interact during programming tasks. By syn-

thesizing these qualitative insights, the study seeks to fill the empirical and conceptual gaps left by previous short-term and quantitatively driven research, ultimately contributing a nuanced explanation of the impact of generative AI on the development of computational thinking in higher education.

2. Literature Review

2.1. Generative Artificial Intelligence on College Students' Computer Thinking

Computational thinking (CT)—a problem-solving approach grounded in abstraction, decomposition, algorithmic design, iteration, and generalization—has long been regarded as a foundational cognitive skill in programming education (Yadav et al., 2016). With the rise of artificial intelligence, scholars increasingly argue that CT is no longer only a programming competency but a core literacy that enables learners to reason with, through, and about AI systems (Asunda et al., 2023; Tlili et al., 2023). This shift has brought renewed attention to how emerging AI tools, especially large language model (LLM)-based generative AI, may transform CT development.

A growing line of studies demonstrates that generative AI can support CT by offering contextual examples, adaptive scaffolding, and personalized feedback. For instance, ChatGPT-supported instruction has been shown to enhance students' CT skills, programming self-efficacy, and motivation (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b). Liao et al. (2024) further identify that ChatGPT can scaffold CT subcomponents—abstraction, decomposition, debugging, and algorithmic thinking—by providing stepwise hints tailored to learners' cognitive needs. Similarly, Hsu (2025) shows that interacting with LLMs allows learners to iteratively test ideas in ways that mirror the exploratory cycles of CT development.

Multiple studies also emphasize motivational gains. Personalized feedback and conversational support can increase engagement and sustain persistence in programming tasks (Karaoglan Yilmaz & Yilmaz, 2022). Experiments indicate that learners in AI-augmented environments show higher confidence and satisfaction, reinforcing the psychological pathways through which CT is developed (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b).

Beyond motivational influences, generative AI supports the acquisition of technical programming skills. Tools such as GPTutor offer detailed explanations and debugging assistance, improving code comprehension and technical performance (Chen et al., 2023). Garcia's (2025) rapid review synthesizes these benefits across contexts—highlighting AI's affordances for individualized tutoring, source-code generation, and automated assessment—while simultaneously cautioning against uncritical adoption.

These findings are aligned with broader evidence showing that programming education often struggles with students' low CT skills, poor self-efficacy, and limited intrinsic motivation (Fagerlund et al., 2021; Tikva & Tambouris, 2021). Gen-

erative AI appears to mitigate some of these barriers by reducing frustration, accelerating problem-solving, and enabling learners to overcome conceptual obstacles more easily.

Despite these advantages, a growing body of research warns that generative AI may also undermine essential components of CT development. Copy-paste behaviours, surface-level engagement with AI-generated code, and increased error rates have been observed when students rely excessively on AI (Sun et al., 2024). Such tendencies can weaken learners' debugging skills, limit metacognitive monitoring, and reduce opportunities for productive struggle—an essential mechanism for developing deeper CT (Zhao et al., 2025).

Similarly, scholars note that over-reliance on AI may impair critical thinking, diminish independent problem-solving, and reduce satisfaction derived from solving problems autonomously (Kazemitabaar et al., 2023). Broader critiques argue that generative AI tools have structural limitations—including hallucinations, formulaic reasoning patterns, and insufficient domain knowledge—that may mislead novice learners (Becker et al., 2023; Boguslawski et al., 2024; Denny et al., 2024a, 2024b).

Thus, while some studies emphasize CT enhancement (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b; Liao et al., 2024), others emphasize cognitive risks and the need for pedagogical safeguards. This mixed pattern reveals an important tension: GAI can both support and hinder CT depending on how students interact with it.

Several design-based studies highlight that structured or guided interactions with generative AI produce more meaningful CT gains. For example, Liao et al. (2024) embed ChatGPT within an explicit CT framework—decomposition, abstraction, algorithmic design—demonstrating targeted improvement of related competencies. Yilmaz & Karaoglan Yilmaz (2023a, 2023b) report similar outcomes when AI is used as a scaffolding tool to support creativity, algorithmic thinking, and cooperation.

Likewise, when prompt design is incorporated explicitly into instruction, students demonstrate deeper reasoning and greater CT refinement (Hsu, 2025). This aligns with work showing that structured prompting produces better programming outcomes than unstructured AI usage (Garg et al., 2025).

Across these studies, CT development through generative AI is shown to be feasible, conditional, and interaction-dependent. In particular, many studies rely on short-term experimental designs and do not investigate how learners experience AI cognitively, emotionally, and socially during complex programming tasks.

2.2. AI Literacy and Computer Thinking

As an emerging dimension of digital competence, AI literacy encompasses the knowledge and skills required to use AI tools accurately, responsibly, and meaningfully. Although AI literacy is widely recognized as a key future competency (Vuorikari, Kluzer, & Punie, 2022), empirical research examining how AI literacy shapes users' behaviors—particularly when interacting with LLM-based systems

such as ChatGPT—remains limited (Pinski & Benlian, 2023). This gap is especially important because LLMs can sometimes produce incorrect, illogical, or fabricated outputs (often described as "hallucinations"), and they currently lack genuine common sense or a grounded understanding of real-world contexts (Ji et al., 2023).

Within this broader framework, computational thinking (CT) has increasingly been viewed as a core component of AI literacy, as it equips learners with the ability to interact with AI technologies effectively, critically, and ethically across both professional and everyday settings (Celik, 2023; Walter, 2024). However, emerging evidence suggests that learners' ability to use generative AI effectively is highly contingent on the design of instructional support. Husain (2024), through interviews with programming instructors, highlights that generative AI frequently provides inaccurate responses and offers limited domain-specific resources, indicating that clearer pedagogical strategies are needed for meaningful teacher–AI and student–AI collaboration (Zhao et al., 2025).

Supporting this concern, Garg et al. (2025) demonstrate that unguided or unstructured use of generative AI can hinder learning outcomes in both data analysis and programming tasks. Students who received structured training in prompt formulation significantly outperformed those who used generative AI without guidance—as well as those who relied solely on traditional internet search methods—underscoring that AI literacy must include explicit instruction in how to engage productively with LLMs.

2.3. Complex Computer Programming and Computer Thinking

Complex programming tasks require learners to navigate high levels of abstraction, uncertainty, and nonlinear problem-solving. Although LLM-based tools can generate code and offer suggestions, research shows that such automated outputs often introduce vulnerabilities, inefficiencies, and logical errors, especially in unfamiliar or sophisticated programming contexts (Hilario et al., 2024; Zhang et al., 2023). When students depend excessively on AI-generated solutions, they may find it difficult to tackle novel problems that demand creative reasoning, strategic planning, and rigorous debugging—skills central to computational thinking.

Debugging is particularly vital in CT development because it requires students to confront errors directly, test hypotheses, iterate on solutions, and refine their conceptual understanding of how code operates (Zhang et al., 2023). These hands-on troubleshooting processes build technical resilience and deepen students' mental models of computational logic. In broader terms, programming education cultivates analytical thinking, logical reasoning, and algorithmic problem-solving—competencies that extend far beyond coding and positively influence academic performance, workplace readiness, and personal growth (Tikva & Tambouris, 2021).

Learning environments that provide immediate, interactive feedback further strengthen these skills. Systems that allow students to identify mistakes in real

time support individualized pacing and help accommodate diverse learning styles (Yilmaz & Karaoglan Yilmaz, 2023a, 2023b; Chen et al., 2023; Tian et al., 2023). However, despite their benefits, generative AI tools such as ChatGPT exhibit critical weaknesses—including limited common sense, potential biases, difficulty handling complex reasoning chains, and an inability to process visual inputs—factors that constrain their reliability in advanced programming scenarios (Rahman & Watanobe, 2023).

Recent evidence also highlights the value of collaborative learning enhanced by LLMs. Yan et al. (2025) found that students engaged in AI-supported collaborative programming experienced significantly reduced cognitive load and showed stronger computational thinking outcomes compared with those in traditional settings. Such findings align with research demonstrating a positive correlation between programming proficiency and CT development, reinforcing the central role of programming education in strengthening CT skills (Belmar, 2022).

Collaborative programming environments help distribute task complexity among team members, facilitating knowledge sharing and reducing cognitive burden during challenging tasks (Zheng et al., 2022; Zhang et al., 2016). Yet, the effectiveness of AI in these settings depends heavily on how learners interact with it. Wang et al. (2024) demonstrated that well-designed prompting strategies substantially improve LLMs' ability to address complex problems. Their work provides a structured framework for educators and highlights the importance of explicit strategy training when integrating LLMs into programming instruction.

Overall, while LLMs offer valuable support for learners navigating complex programming work, their inconsistency, occasional inaccuracies, and limited reasoning capabilities underscore the need for ongoing human oversight and robust foundational knowledge (Holmes et al., 2022; Usher, 2025). These insights suggest that AI assistance should complement—not replace—students' development of core computational thinking competencies.

2.4. Constructionism and Computational Thinking

Constructionism provides an important theoretical lens for understanding how computational thinking (CT) develops in AI-supported learning environments. Rooted in Papert's (1980) original formulation, constructionism emphasizes that learning should be active, personal, and experiential, rather than a passive reception of information. Through hands-on creation and iterative problem-solving, learners gain ownership over their learning processes and develop deeper forms of understanding. Later work affirms that constructionist learning environments foster not only CT elements—such as abstraction, debugging, and algorithmic thinking—but also a sense of responsibility and engagement that makes learning more meaningful (Csizmadia et al., 2019).

Building on this view, recent research suggests that generative AI can function as a powerful constructionist partner. Zhao et al. (2025) show that when students actively communicate and interact with generative AI to construct solutions, they

demonstrate improved learning achievement. This aligns with social constructivist perspectives, which position LLMs as mediating tools that facilitate dialogue, peer interaction, and shared meaning-making—factors known to promote CT development (Sharma et al., 2019). In AI-supported collaborative environments, students often spend more time discussing strategies, refining algorithms, and negotiating problem-solving pathways, which ultimately strengthens their computational thinking. Evidence from experimental interviews further confirms that students trained to use AI collaboratively allocate more cognitive effort to algorithmic reasoning than those in conventional settings (Yan et al., 2025).

From a constructionist standpoint, CT itself can be understood as a process of learning through creation, iteration, and reflection. Lodi and Martini (2021) note that Papert envisioned CT as emerging through direct interaction with computational artifacts, particularly programming tasks that require learners to externalize and test their ideas. Generative AI tools extend this interaction by enabling students to explore new concepts ("exploration" pattern) or accelerate progress toward solutions ("acceleration" pattern), as identified by Barke et al. (2023). These distinct patterns show that learners engage with AI differently depending on their goals, and such engagement can meaningfully shape their CT practices.

Affective factors also play a central role in constructionist learning. Previous studies show that students' attitudes toward programming strongly predict CT development and learning achievement (Sun et al., 2020). Generative AI environments may enhance these attitudes by providing rich technological experiences that foster intrinsic motivation, persistence, and sustained engagement in programming activities (Choi & Kim, 2024; Zhao et al., 2025). Moreover, integrating AI into instructional settings has been shown to boost self-efficacy, as intelligent feedback systems and virtual assistants offer real-time guidance that reinforces learners' confidence and supports positive learning trajectories (Kim & Kim, 2024; Yang et al., 2024).

These findings suggest that when framed through constructionism and social constructivism, generative AI can serve as both a cognitive tool and a motivational catalyst, creating environments in which learners actively build knowledge, refine computational thinking skills, and develop stronger learning dispositions. Based on the above analysis, this study proposes the following research questions:

Research Question 1 (RQ1): How do college students cognitively engage with generative AI tools when solving complex computer programming tasks, particularly in the development of computational thinking skills?

Research Question 2 (RQ2): How do students emotionally respond to generative AI assistance during complex programming tasks, and how do these emotional experiences influence their computational thinking and self-efficacy?

Research Question 3 (RQ3): How do students negotiate and adapt their behavioral strategies—such as prompt design, debugging practices, and collaboration—when working with generative AI in complex programming tasks?

3. Methods

3.1. Research Design

This study employs a qualitative research design to explore the impact of generative artificial intelligence on college students' computer thinking in the task of complex computer programming based on social cognitive theory. In-depth semi-structured interviews serve as the primary data collection method, allowing for the exploration of students' cognitive processes, emotional experiences, and behavioral strategies when using generative AI to complete complex programming tasks.

3.2. Pilot Study

In a pilot study, two participants were included to assess their comprehension, ease of use, and clarity of the interview protocol, thereby informing the development of appropriate research questions. Based on the results of the pilot study, the research team revised the wording. The participant noted that the "Self-efficacy" is not an easy term to understand, so replace it with "Self-confidence". For Question 14: "How many rounds of dialogue do you think are typically required between you and a generative AI to resolve a problem?" Participants indicated that a specific number of rounds could not be determined, as this depends on the complexity of the task and necessitates iterative adjustments until an accurate outcome is achieved. Consequently, this question was excluded from the interview question framework. In general, the interview questions were generally adapted to be more culturally relevant and understandable.

3.3. Recruitment of Participants

This study recruited sophomore students majoring in Computer Science at Inner Mongolia Vocational College of Mechanical and Electrical Engineering, who were enrolled in the course Front-end Programming Design and Development, as participants. A 15-week instructional experiment was conducted, involving 48 students in the experimental group and 47 in the control group. The experimental group received AI tool-supported learning, whereas the control group engaged in traditional teacher-guided learning. Prior to the experiment, the experimental group underwent training on AI tool operation to facilitate their rapid and accurate utilization of AI for programming-assisted learning activities. Among the 48 students in the experimental group, 16 volunteered to participate in this study, consisting of 10 females and 6 males.

3.4. Participants

As Table 1 shows, a total of 16 college students were interviewed, comprising ten females and six males, with ages ranging from 19 to 21. Their major is Computer Application Technology.

Having obtained consent, the researcher conducted semi-structured interviews lasting between 30 and 45 minutes. Each interview was audio-recorded to gather

qualitative data, and these recordings were subsequently transcribed.

Table 1. Basic information table of the interviewees.

Code	Gender	Age	Major (Present)
Interviewee 01	Male	19	Computer Application Technology
Interviewee 02	Male	20	Computer Application Technology
Interviewee 03	Female	20	Computer Application Technology
Interviewee 04	Female	19	Computer Application Technology
Interviewee 05	Female	21	Computer Application Technology
Interviewee 06	Female	21	Computer Application Technology
Interviewee 07	Female	19	Computer Application Technology
Interviewee 08	Male	20	Computer Application Technology
Interviewee 09	Female	20	Computer Application Technology
Interviewee 10	Female	20	Computer Application Technology
Interviewee 11	Female	20	Computer Application Technology
Interviewee 12	Female	20	Computer Application Technology
Interviewee 13	Male	20	Computer Application Technology
Interviewee 14	Female	20	Computer Application Technology
Interviewee 15	Male	20	Computer Application Technology
Interviewee 16	Male	20	Computer Application Technology

3.5. Thematic Analysis

This study employed the six-step thematic analysis approach developed to analyze the transcribed semi-structured interview data collected from 16 undergraduate students majoring in Computer Application Technology. Initially, the research team conducted repeated readings of the interview transcripts to identify and code key information pertinent to applications based on Generative Artificial Intelligence, encompassing application methods, user experiences, and impacts on programming competence, which resulted in the generation of 20 initial codes. Subsequently, these initial codes were categorized and integrated through semantic correlation analysis. To refine these themes, the research team verified the relevance between each theme and the original data, eliminated redundant categories, and clarified thematic boundaries.

3.6. Ethical Considerations

This study strictly adhered to ethical standards throughout the entire research process, and fully safeguarded the rights, dignity and privacy of all research participants. All participants signed a written informed consent form. Interview data were processed using anonymous codes (e.g., P01, P02), and audio recordings, written transcripts and analytical materials were used exclusively for the research team. All data were stored in encrypted devices or password-protected folders.

4. Findings

4.1. Theme 1: Cognitive Processing: Reshaping Task Understanding, Problem Decomposition, and Logical Reasoning with AI Involvement

Among the 16 respondents in this study, almost all showed significant changes in cognitive processing during their interactions with generative AI, particularly in task comprehension, problem breakdown, code logic reasoning, and verification awareness. Overall, AI served as a cognitive scaffold, helping students build structured thinking processes, and also fostered a deeper understanding of computational logic through a cycle of comparison, debugging, and verification.

First, AI facilitates a more structured approach to complex tasks for students. Many students, when starting programming tasks, adopt a "analyze the task first—describe it to the AI—and then have the AI refine it" model. For example, Respondent 01 emphasized that she would "think about the approach herself first, and then tell the AI the steps," naturally breaking down the task into several substeps, and then assisting the AI in producing a more accurate solution.

Secondly, students have developed a more sophisticated problem analysis and verification mechanism than before through interaction with AI. Respondent 08 described their operation in detail: "Paste my own code and compare it with the AI's approach to find the differences." This "differential debugging" has become a common strategy adopted by many students, demonstrating the improved comparative reasoning ability brought about by AI.

Furthermore, AI has facilitated a deeper understanding of logical relationships among students. Respondent 15 pointed out that after interacting with AI, their thinking "became more structured, focused more on solution verification, and better understood the essence of logic," demonstrating a cognitive upgrade from "writing code" to "understanding why the code works the way it does."

Fourth, AI guides students to develop a cyclical debugging and improvement mindset. Respondent 12 stated that they would "run → check the error message → find the cause → ask the AI again → run again," demonstrating a continuously iterative verification cycle. In this process, students gradually master how to locate errors and adjust strategies based on feedback, which is highly consistent with the "debugging mindset" in classic computational thinking (CT).

Furthermore, AI enhanced students' awareness of "multiple-solution thinking." Respondent 10 pointed out that AI can provide multiple solutions for comparison, prompting them to actively think about "which solution is better," rather than passively accepting a single solution. This multiple-solution reasoning model significantly promotes students' abstract thinking and strategy selection abilities, a learning mechanism rarely presented in traditional classrooms.

Finally, the cognitive support provided by AI is not entirely positive. Some students mentioned that AI makes understanding easier, but it may also weaken independent thinking. For example, respondents 14 and 15 both realized that if they directly adopted the complete code provided by AI, they would "skip thinking

about the details." This indicates that while AI enhances cognitive processing, it may also cause a "cognitive substitution effect," meaning that students reduce their independent reasoning and error-finding processes.

In summary, AI's impact on cognitive processing exhibits a "two-way effect": on the one hand, it significantly enhances students' computational thinking abilities through task decomposition, structured expression, difference comparison, and cyclical verification; on the other hand, it may also weaken students' independent reasoning, making them prone to relying on AI's direct output in certain situations. Students' cognitive strategies, with the participation of AI, are evolving into a "human-machine hybrid cognitive structure," retaining human judgment while embedding AI-generated logical frameworks—one of the most important cognitive findings of this study.

4.2. Theme 2: Emotional Experience: Fluctuations in Self-Efficacy, Emotion Regulation, and Reshaping of Learning Motivation in AI Interaction

With the intervention of generative AI, students' emotional experiences during programming learning are highly diverse and dynamic. Overall, students' emotional responses mainly revolve around fluctuations in self-efficacy, perceptions of AI reliability, frustration and anxiety, excitement and a sense of accomplishment from success, and ambivalent emotions arising from long-term dependence. According to Marcionetti and Castelli (2023), self-efficacy functions as a central mechanism within a social cognitive model, linking other factors to the explanation of job satisfaction and life satisfaction. This topic reveals how AI profoundly influences students' emotional structures and learning experiences beyond cognition.

First, the experience of success is the most common and direct source of emotion for students. Many students mentioned that they immediately felt a strong sense of accomplishment when the code provided by the AI could "run successfully on the first try." For example, Respondent 14 said that after successfully running the code output by the AI, "I felt that the task was progressing smoothly, which made me more confident." This immediate positive feedback significantly enhanced their learning motivation.

However, the unpredictability of AI also brings considerable frustration and anxiety. Especially during the debugging phase, if the AI's answers repeatedly fail, students' emotions quickly turn negative. For example, interviewee 15 described the most typical emotional drop: "The AI made more and more mistakes each time, constantly giving wrong answers, it was worse than not asking at all." This continuous failure made him doubt the AI's capabilities and caused temporary anxiety.

It's worth noting that some students, when faced with AI errors, attribute the problem to the AI rather than themselves, thus minimizing the negative emotional impact. When asked whether AI errors affected their self-confidence, Respondent

16 explicitly answered "no," because the task had to be completed, and AI was merely one of the tools. This "external attribution" reduces the risk of decreased self-efficacy and reflects students' gradual development of a correct understanding of AI—that AI errors do not equate to their own failure.

At the same time, the involvement of AI has also brought students a significant sense of security and emotional comfort. Respondent 07 believes that AI is a "guide who is always available," allowing them to seek help immediately when encountering difficulties, effectively reducing feelings of confusion.

On the other hand, the more powerful AI becomes, the more likely it is to create a kind of "psychological dependence" among students. For example, interviewee 10 mentioned that whenever they encounter difficulties, their "first reaction is to ask AI" because AI can always quickly provide direction. While this dependence brings immediate emotional relief, it also creates a sense of contradiction for some students in the long run—they enjoy the ease brought by AI, but also worry that their own abilities will be weakened.

Finally, the role of AI in group collaboration directly impacts students' emotional experience. Later, when students discovered that AI could provide more objective and faster solutions within the team, arguments decreased and efficiency increased, leading many students to feel "relaxed" and "smooth." Respondent 05 pointed out that with AI, group collaboration "was no longer about guessing each other's questions, but about verifying AI's solutions together," significantly reducing communication pressure.

Overall, students' emotional experiences with AI support exhibit a significant duality. On the one hand, AI brings a sense of accomplishment, efficiency, security, and ease of understanding; on the other hand, its erroneous output, uncontrollability, and potential dependency bring anxiety, frustration, and conflicting emotions. This emotional fluctuation is not simply a matter of positive or negative, but rather occurs alternately at different stages of the learning process, constituting a unique "emotional trajectory" for learners in the AI era. Through these emotional dynamics, we can see that generative AI not only changes the way students learn but also reshapes their emotional relationship with tasks, failures, and successes.

4.3. Theme 3: Behavioral Strategies: Hybrid Problem-Solving Paths and Learning Operation Modes Formed in Human-AI Interactions

With the deep integration of generative AI, students have gradually developed a set of behavioral strategies with universal regularities and human-machine collaboration characteristics when completing programming tasks.

These strategies not only reflect students' adaptive use of AI tools but also the process of reconstructing their learning logic. Overall, students' behavioral strategies exhibit a multi-stage path of independent attempt—AI assistance—human verification—cyclical debugging, accompanied by continuously maturing prompt

word construction capabilities and debugging methods.

First, almost all respondents adopted a two-stage strategy of "trying independently first, then seeking help from AI." Respondent 16 explicitly stated that they were accustomed to "using their own knowledge first, and then using AI assistance if they couldn't solve it," indicating that AI is a secondary problem-solving tool that intervenes after independent analysis fails.

Secondly, many students have developed increasingly sophisticated prompting strategies in their interactions with AI. Respondent 15's approach is the most representative; he "establishes a new conversation, introduces his role, explains the task, and gets the AI to think carefully," demonstrating a systematic understanding of role setting, supplementing task context, and providing guiding prompts.

Third, with the support of AI, students have developed a highly cyclical debugging behavior path. Respondent 12 described a typical process: "Run → See error message → Ask AI → Modify → Run again", and this cycle appeared in almost all respondents.

Fourth, with the help of AI, students have begun to use a "comparison strategy" to understand code logic and improvement plans. Respondent 08 pointed out that he would compare his own code with the AI-generated code to find the causes of errors from the differences. This difference analysis not only helps students locate problems but also strengthens their structured thinking.

Fifth, the involvement of AI has also driven a shift in students' behavioral strategies towards "multi-solution exploration." Respondent 10 pointed out that AI provides multiple feasible solutions, allowing them to choose the most suitable one, rather than relying on a single method. This "solution selection strategy" reflects students' gradual shift from linear thinking to diversified strategic thinking, something difficult to achieve in traditional teaching.

Furthermore, students' behavioral strategies in group collaboration have changed significantly with the addition of AI. Respondent 15 explained that AI transformed group collaboration "from guessing line by line to comparing AI's solutions," reducing unnecessary arguments and improving efficiency; Respondent 05 also pointed out that AI can quickly provide ideas, making discussions more focused on logical aspects rather than basic errors. This shows that AI has become an "external intellectual resource" in team collaboration.

However, these positive trends coexist with potential risks. Several students (such as respondents 14 and 15) mentioned that the efficiency of AI might lead to skipping independent reasoning steps and focusing only on the final answer. Therefore, students' optimization of behavioral strategies includes both "more efficient and more structured" and "more reliant and less effort."

Overall, students developed a complex and dynamic system of behavioral strategies through their interaction with AI. These strategies included independent experimentation, prompt optimization, iterative debugging, code comparison, and multiple solution selection, as well as strategy adjustments in teamwork. These strategies not only demonstrate students' adaptive growth with AI tools but also reveal how the human-AI symbiotic learning model reshapes students' behavioral

structures. From a behavioral perspective, learners gradually entered a "hybrid problem-solving system," relying on AI's rapid processing capabilities while retaining and strengthening their own logical judgment and strategy selection abilities.

4.4. Theme 4: Human-AI Collaboration Methods: Dynamic Division of Labor, Cognitive Complementarity, and Restructuring of Learning Processes

Among the 16 respondents in this study, the human-AI collaboration exhibited a distinctly structured characteristic. Students not only learned how to use AI, but also developed a stable "collaborative mental model" during the learning process: humans were responsible for goal setting, judgment, and integration, while AI was responsible for generating solutions, debugging, and interpreting them. This collaborative model was not static, but dynamically adjusted between task difficulty, student comprehension, and AI performance, demonstrating a human-AI joint problem-solving approach with practical wisdom.

First, the collaborative relationship is based on human dominance. Many students explicitly stated that they played the roles of "decision-maker" and "task planner" in the collaborative process. For example, Respondent 15 emphasized: "I led most of the time, and the AI assisted me; I only needed to provide the key code." This indicates that students did not allow the AI to completely replace their own judgment, but rather regarded the AI as an "intelligent assistant that can be invoked on demand."

Secondly, the core characteristic of human-AI collaboration lies in the clear division of roles. Respondent 14 summarized their collaboration model in one sentence: "I am the demand decomposer, and AI is the solution provider and problem solver." This statement not only presents the specific collaboration chain, but also reveals the important position of AI in task completion—it is neither a passive tool nor a full agent, but an intelligent partner that undertakes execution, generation, and local reasoning.

Third, a key value of AI in collaboration lies in providing "externalized thinking and multiple solution options." When serving group tasks, AI acts as a "third-party proposer." Respondent 10 pointed out that the role of AI in group collaboration is: "Let the AI provide several solutions, and then the group chooses the most suitable one." This approach clearly changes the traditional collaboration model, shifting the focus of collaboration from "identifying problems" to "evaluating solutions," improving the efficiency of team discussions, and making thinking more strategic and selective.

Fourth, the efficiency improvement of human-AI collaboration is particularly significant, especially in debugging and error localization. Respondent 15 believes that with AI, the team no longer needs to spend time checking code line by line; instead, they can "hand it over to AI, and it's crystal clear." AI plays a role more akin to an "automatic debugging tool" or "rapid technical consultant" among professional engineers, significantly reducing collaboration costs and freeing up stu-

dents' cognitive resources, allowing them to focus their energy on higher-level logical reasoning or solution design.

Fifth, "dynamic role switching" also occurs during the collaboration process. When students encounter basic errors, they usually maintain the leading role; however, when the problem becomes complex, 16 respondents indicated that they would "hand over the initiative to the AI, letting it propose ideas first, and then judge whether they are feasible." This shows that the collaboration model is not static, but automatically adjusts the division of labor according to the complexity of the task to achieve more efficient problem-solving. The human-AI interaction thus presents a "flexible collaboration model": when students have sufficient knowledge, humans take the lead; when the problem exceeds their capabilities, AI takes the lead.

Sixth, AI can also act as an "interpreter" in collaborations, helping students to understand code and logic in reverse. For example, Respondent 15 said that he not only asks AI to modify code, but also "asks it why it does it this way," treating AI as a "reasonable partner." This role transcends traditional instrumental functions, making AI a "conversational source of knowledge," which plays an important role in promoting students' understanding and reflective abilities.

However, human-AI collaboration is not always smooth. Many students mentioned that AI is often powerless when encountering version differences, framework differences, or real-world engineering scenarios. For example, respondents 15 and 16 both pointed out that AI "cannot handle version compatibility issues," indicating that in human-AI collaborative systems, humans still need to bear the responsibility for environmental judgment and final decision-making. Environmental factors is a basic element in Social Learning Theory. A key idea of Bandura's (1977) Social Learning Theory is reciprocal determinism, which comprises three elements that affect behavior: the individual (P), the environment (E), and the behavior itself.

Overall, the human-AI collaboration model presents a highly practical and flexible complementary structure. Students lead the overall direction and judgment, while AI provides solution generation, debugging, and interpretation, forming a closed loop for collaborative problem-solving. In this collaborative relationship, the learner's role shifts from the traditional "executor" to "commander and evaluator," while AI evolves from a tool to an "external cognitive partner," jointly constructing a new, dynamic learning ecosystem characterized by knowledge sharing. Cognitive partner is the use of AI or technology as instruments for group learning. By providing tailored feedback, encouraging conversation, and fostering involvement, cognitive partners promote learning (Cain, 2023; Salomon & Perkins, 2005).

4.5. Theme 5: Limitations and Risks of AI: The Dual Dilemma of Capability Boundaries, Error Accumulation, and Cognitive Dependence

While generative AI provides strong support in the learning process, respondents'

accounts also indicate that the application of AI in programming education has significant limitations and risks. These risks mainly involve three aspects: unsolvable problems due to technological limitations, learning interference and cognitive confusion caused by erroneous output, and potential dependency and weakening of thinking due to long-term use. These phenomena collectively constitute the limiting aspects of AI-assisted learning, forming a stark contrast with the positive effects mentioned earlier, and revealing the complexity of human-AI collaboration.

First, the most common and prominent problem is that AI cannot handle compatibility and contextual dependencies in real-world engineering environments. For example, Respondent 15 emphasized that AI has a significant deficiency in understanding real-world development environments: "For example, HarmonyOS, WeChat Mini Program development... There will be version compatibility issues that AI cannot solve."

Secondly, AI often exhibits logical inconsistencies or recurring errors, which directly disrupts students' learning pace and emotional stability. For example, respondent 15 described a typical frustrating experience: "The AI makes more and more mistakes each time it runs, constantly reporting errors; it's worse than not asking at all." This "accumulated error effect" not only increases students' confusion but also negatively impacts their ability to self-debug.

Third, the lag in knowledge updates and limited data sources in AI often lead to solutions that "seem correct but are unusable." Respondent 14 pointed out: "AI cannot solve innovative problems or problems without publicly available examples." AI's performance is particularly limited in situations requiring creative thinking or adherence to the latest development standards. Students therefore must be able to judge which tasks can rely on AI and which require independent reasoning or consulting authoritative documentation, which is an additional challenge for beginners.

A deeper risk stems from over-reliance, a phenomenon many respondents expressed concern about. Respondents 14 and 15 both pointed out that if students rely on AI to generate complete code for an extended period, they will "skip thinking about details" and gradually lose their ability to reason independently.

Furthermore, while AI's explanatory capabilities are convenient, they can sometimes create an "illusion of understanding." Respondent 05 pointed out that sometimes AI explanations "sound reasonable, but don't actually work," leading students to mistakenly believe they have understood. This "overly credible explanation" makes it difficult for students to distinguish between genuine understanding and superficial acceptance, potentially leading to flawed knowledge structures.

On the other hand, while the "multiple solutions" of AI can promote divergent thinking, it can also lead to decision-making difficulties and information overload. Respondent 10 mentioned that after AI provided multiple solutions, they "didn't know which one to choose" and needed to spend extra time verifying each method. This suggests that excessive AI generation may create new burdens, caus-

ing learning efficiency to decrease rather than increase.

In group collaborations, the involvement of AI may also lead to a weakening of abilities due to role changes. Some students have begun to rely on AI to generate solutions, resulting in a decrease in the depth of discussions within the team. For example, interviewee 15 mentioned that group members often "directly follow what the AI gives," reducing logical discussions among themselves, which may weaken the team's collaborative reasoning ability in the long run.

Overall, the limitations and risks of AI are multi-layered:

(1) Limitations at the tool level: inability to handle environmental dependence, version compatibility, and innovation issues;

(2) Risks at the cognitive level: error accumulation, misunderstanding, and over-reliance;

(3) Side effects at the learning behavior level: reduced independent thinking, decreased spirit of exploration, and reduced depth of team collaboration.

These limitations remind us that AI is not a perfect teaching tool; its educational value depends on learners' ability to maintain critical judgment during use and avoid transforming from a "facilitator" into a "substitute." From a research perspective, these risks not only reveal the boundaries of AI-supported learning but also emphasize the importance of cultivating students' metacognitive abilities, enabling them to effectively manage human-AI collaborative relationships, maintain independent thinking, and prevent the long-term weakening of cognitive abilities.

5. Discussions and Conclusions

5.1. Overall Interpretation of Findings through Social Cognitive Theory

The findings of this study reveal that generative AI is not merely a technical aid but a deeply embedded cognitive, emotional, and behavioral scaffold that reshapes how college students engage with complex programming tasks. When interpreted through the lens of Social Cognitive Theory (Bandura, 1986), students' experiences show clear evidence of triadic reciprocal determinism: personal factors, environmental inputs, and behavioral strategies continuously interact throughout the human-AI collaboration process. At the cognitive level, AI reshaped learners' problem-solving pathways by facilitating task decomposition, structured reasoning, and iterative debugging, yet simultaneously introduced risks of cognitive substitution, aligning with SCT's emphasis on metacognitive self-regulation.

Eliciting everything from increased confidence and reduced frustration to anxiety caused by error buildup, AI demonstrates SCT's assertion that emotional states serve as key mediators of self-efficacy. Students could be seen adopting hybrid strategies that intertwined independent work with AI verification, signaling behavioral adaptation influenced by environmental feedback. Revealed through dynamic role negotiation and cognitive complementarity is a human-AI collaboration model in which learners come to see AI as an "external cognitive partner," widening SCT's traditional observational learning framework. The mechanisms

by which computational thinking is constructed, negotiated, and internalized are shown to be significantly transformed by generative AI, while structural risks are exposed that require stronger pedagogical and AI literacy supports.

5.2. Compare the Findings of This Study with Previous Research

Prior studies widely demonstrate that generative AI can support CT development, particularly by facilitating abstraction, decomposition, algorithmic reasoning, and debugging. Experimental work by Yilmaz & Karaoglan Yilmaz (2023a, 2023b), Gong et al. (2025), and Zhao et al. (2025) shows significant improvements in CT skills and programming performance when AI-supported instruction is compared with traditional methods. Similarly, Hsu (2025) and Ponzini et al. (2024) emphasize that ChatGPT aids students by providing contextual explanations, stepwise guidance, and clear task breakdowns.

These findings resonate strongly with the present study: students described AI as a *cognitive scaffold* that supports task decomposition, logic clarification, iterative debugging, and multi-solution comparison. However, unlike controlled experiments, this study offers fine-grained qualitative evidence of how these cognitive benefits unfold in real problem-solving cycles, capturing students' moment-to-moment reasoning and verification behaviors.

Consistent with experimental research, such as Yilmaz & Karaoglan Yilmaz (2023a, 2023b) and Zhao et al. (2025), this study found that generative AI increases learners' confidence, reduces frustration, and enhances motivation. Participants frequently described AI as a "safety net," mirroring previous claims that AI can lower barriers to engagement for novice programmers.

Unlike prior work—which mainly reports *net motivational gains*—this study reveals the emotional volatility of AI-supported learning. Learners oscillated between satisfaction and anxiety depending on AI accuracy, highlighting nuanced emotional trajectories not captured in previous surveys or experiments.

Several scholars—particularly Yilmaz & Karaoglan Yilmaz (2023a, 2023b) and Kohen-Vacs et al. (2025)—warn that ChatGPT may diminish independent thinking, reduce critical reasoning, or produce misleading outputs. The present study strongly corroborates these concerns: students reported over-reliance, illusion of understanding, and the erosion of self-initiated problem solving when AI outputs became the default starting point.

However, this study goes further by identifying mechanisms of cognitive substitution, such as how students skip reasoning steps, attribute mistakes externally, or prematurely accept AI explanations. These mechanisms clarify *how* dependence develops, offering richer insight than prior questionnaire-based findings.

Previous studies mention improved debugging or problem-solving efficiency but do not conceptualize collaboration structures. Findings here reveal distinct human-AI role configurations: students acting as planners/evaluators while AI serves as generator/debugger/explainer. This dynamic, fluid division of labor—absent from the existing literature—extends the understanding of AI-supported

learning toward a hybrid cognitive agency model.

Additionally, while Gong et al. (2025) document interaction patterns under generative prompting, the detailed behavioral strategies uncovered in this study (e.g., verification loops, difference-comparison debugging, role-setting prompt techniques) provide a far more granular view of how students operationalize AI in authentic programming tasks.

Most existing studies rely on short-term interventions, which capture immediate skill gains but overlook sustained developmental trajectories. This study reveals unresolved issues—such as long-term dependence, shifting self-efficacy, and changes in metacognitive regulation—that prior studies do not empirically examine.

Furthermore, whereas prior research focuses primarily on outcomes (CT scores, task performance), the present findings highlight process-level tensions: how AI reshapes learners' internal reasoning, their emotional resilience toward failure, and their negotiation of agency in collaborative contexts.

Overall, this study confirms the cognitive and motivational benefits of AI identified in earlier research while simultaneously exposing structural risks that prior studies only speculated about. Thus, while prior work portrays AI largely as an instructional enhancer, this study demonstrates that AI simultaneously operates as a cognitive partner, motivational force, and potential source of epistemic risk—requiring more sophisticated pedagogical frameworks for responsible integration.

5.3. Theoretical Contributions

This study makes several theoretical contributions to the emerging scholarship on AI-supported learning and computational thinking development. By integrating generative AI into the Social Cognitive Theory framework, SCT is extended beyond its traditional human-centered scope, illustrating that non-human agents can act as key mediators of cognitive, emotional, and behavioral regulation. It is not only informational feedback that AI offers, but also support for learners' self-efficacy development, emotional regulation, and strategic modeling—responsibilities traditionally held by instructors and peers. The study enhances computational thinking theory by empirically demonstrating AI's scaffolding of abstraction, decomposition, iterative debugging, and diverse-solution reasoning. What distinguishes this study from prior work is its portrayal of AI not as an automation tool but as a cognitive partner whose explanatory, comparative, and iterative processes reconstruct the epistemic foundations of programming learning.

5.4. Limitations and Future Research

Despite its contributions, this study has several limitations that open avenues for future research. First, although considered sufficient for qualitative inquiry, the sample size is potentially limited in generalizability because it was taken from one particular course. Future research should integrate mixed-method or longitudinal approaches, and doing so will allow scholars to observe the temporal dynamics of

AI-mediated learning behaviors. Second, while participants reported both benefits and risks of AI use, the long-term cognitive implications—such as sustained reliance, shifts in self-efficacy, or changes in computational thinking skills—remain unknown. Longitudinal cognitive assessments would be valuable in examining whether AI's support leads to lasting skill enhancement or gradual erosion of independent problem-solving. Finally, ethical considerations such as academic integrity, transparency of AI contributions, and equitable access deserve deeper exploration as AI becomes increasingly integrated into educational ecosystems. Addressing these limitations will be essential for developing comprehensive theoretical and pedagogical frameworks that guide AI's responsible use in computing education.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

Asunda, P., Faezipour, M., Tolemy, J., & Engel, M. (2023). Embracing Computational Thinking as an Impetus for Artificial Intelligence in Integrated STEM Disciplines through Engineering and Technology Education. *Journal of Technology Education, 34,* 43-63. https://doi.org/10.21061/jte.v34i2.a.3

Bandura, A. (1977). *Social Learning Theory.* Prentice Hall.

Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded Copilot: How Programmers Interact with Code-Generating Models. *Proceedings of the ACM on Programming Languages, 7,* 85-111. https://doi.org/10.1145/3586030

Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). Programming Is Hard- or at Least It Used to Be: Educational Opportunities and Challenges of AI Code Generation. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (pp. 500-506). ACM. https://doi.org/10.1145/3545945.3569759

Belmar, H. (2022). Review on the Teaching of Programming and Computational Thinking in the World. *Frontiers in Computer Science, 4,* Article ID: 997222. https://doi.org/10.3389/fcomp.2022.997222

Bandura, A. (1986). *Social Foundations of Thought and Action: A Social Cognitive Theory.* Prentice Hall.

Boguslawski, S., Deer, R., & Dawson, M. G. (2024). Programming Education and Learner Motivation in the Age of Generative AI: Student and Educator Perspectives. *Information and Learning Sciences, 126,* 91-109. https://doi.org/10.1108/ils-10-2023-0163

Cain, W. (2023). Prompting Change: Exploring Prompt Engineering in Large Language Model AI and Its Potential to Transform Education. *TechTrends, 68,* 47-57. https://doi.org/10.1007/s11528-023-00896-0

Celik, I. (2023). Exploring the Determinants of Artificial Intelligence (AI) Literacy: Digital Divide, Computational Thinking, Cognitive Absorption. *Telematics and Informatics, 83,* Article 102026. https://doi.org/10.1016/j.tele.2023.102026

Chen, C., & Chung, H. (2024). Fostering Computational Thinking and Problem-Solving in Programming: Integrating Concept Maps into Robot Block-Based Programming. *Journal of Educational Computing Research, 62,* 186-207.

https://doi.org/10.1177/07356331231205052

Chen, E., Huang, R., Chen, H., Tseng, Y., & Li, L. (2023). GPTutor: A ChatGPT-Powered Programming Tool for Code Explanation. In N. Wang, G. Rebolledo-Mendez, V. Dimitrova, N. Matsuda, & O. C. Santos, (Eds.), *Communications in Computer and Information Science* (pp. 321-327). Springer. https://doi.org/10.1007/978-3-031-36336-8_50

Choi, S., & Kim, H. (2024). The Impact of a Large Language Model-Based Programming Learning Environment on Students' Motivation and Programming Ability. *Education and Information Technologies, 30,* 8109-8138. https://doi.org/10.1007/s10639-024-13107-x

Csizmadia, A., Standl, B., & Waite, J. (2019). Integrating the Constructionist Learning Theory with Computational Thinking Classroom Activities. *Informatics in Education, 18,* 41-67. https://doi.org/10.15388/infedu.2019.03

Denny, P., Leinonen, J., Prather, J., Luxton-Reilly, A., Amarouche, T., Becker, B. A. et al. (2024a). Prompt Problems: A New Programming Exercise for the Generative AI Era. *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* (pp. 296-302). ACM. https://doi.org/10.1145/3626252.3630909

Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J. et al. (2024b). Computing Education in the Era of Generative Ai. *Communications of the ACM, 67,* 56-67. https://doi.org/10.1145/3624720

Eteng, I., Akpotuzor, S., Akinola, S. O., & Agbonlahor, I. (2022). A Review on Effective Approach to Teaching Computer Programming to Undergraduates in Developing Countries. *Scientific African, 16,* e01240. https://doi.org/10.1016/j.sciaf.2022.e01240

Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2021). Computational Thinking in Programming with Scratch in Primary Schools: A Systematic Review. *Computer Applications in Engineering Education, 29,* 12-28. https://doi.org/10.1002/cae.22255

Garcia, M. B. (2025). Teaching and Learning Computer Programming Using ChatGPT: A Rapid Review of Literature Amid the Rise of Generative AI Technologies. *Education and Information Technologies, 30,* 16721-16745. https://doi.org/10.1007/s10639-025-13452-5

Garg, A., Nisumba Soodhani, K., & Rajendran, R. (2025). Enhancing Data Analysis and Programming Skills through Structured Prompt Training: The Impact of Generative AI in Engineering Education. *Computers and Education: Artificial Intelligence, 8,* Article 100380. https://doi.org/10.1016/j.caeai.2025.100380

Gong, X., Xu, W., & Qiao, A. (2025). Exploring Undergraduates' Computational Thinking and Human-Computer Interaction Patterns in Generative Progressive Prompt-Assisted Programming Learning. *International Journal of Educational Technology in Higher Education, 22,* Article No. 51. https://doi.org/10.1186/s41239-025-00552-y

Hilario, E., Azam, S., Sundaram, J., Imran Mohammed, K., & Shanmugam, B. (2024). Generative AI for Pentesting: The Good, the Bad, the Ugly. *International Journal of Information Security, 23,* 2075-2097. https://doi.org/10.1007/s10207-024-00835-x

Holmes, W., Porayska-Pomsta, K., Holstein, K., Sutherland, E., Baker, T., Shum, S. B. et al. (2022). Ethics of AI in Education: Towards a Community-Wide Framework. *International Journal of Artificial Intelligence in Education, 32,* 504-526. https://doi.org/10.1007/s40593-021-00239-1

Hsu, H. (2025). From Programming to Prompting: Developing Computational Thinking through Large Language Model-Based Generative Artificial Intelligence. *TechTrends, 69,* 485-506. https://doi.org/10.1007/s11528-025-01052-6

Husain, A. (2024). Potentials of ChatGPT in Computer Programming: Insights from Pro-

gramming Instructors. *Journal of Information Technology Education: Research, 23,* Article 002. https://doi.org/10.28945/5240

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y. et al. (2023). Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys, 55,* 1-38. https://doi.org/10.1145/3571730

Karaoglan Yilmaz, F. G., & Yilmaz, R. (2022). Examining Student Views on the Use of the Learning Analytics Dashboard of a Smart MOOC. *International Azerbaijan Congress on Life, Social, Health, and Art Sciences*.

Kazemitabaar, M., Chow, J., Ma, C. K. T., Ericson, B. J., Weintrop, D., & Grossman, T. (2023). Studying the Effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. In A. Schmidt, K. Väänänen, T., P. O. Kristensson, A. Peters, S. Mueller, J. R. Williamson, & M. L. Wilson (Eds.), *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (pp. 1-23). ACM. https://doi.org/10.1145/3544548.3580919

Kim, B., & Kim, M. (2024). The Influence of Work Overload on Cybersecurity Behavior: A Moderated Mediation Model of Psychological Contract Breach, Burnout, and Self-Efficacy in AI Learning Such as ChatGPT. *Technology in Society, 77,* Article 102543. https://doi.org/10.1016/j.techsoc.2024.102543

Kohen-Vacs, D., Usher, M., & Jansen, M. (2025). Integrating Generative AI into Programming Education: Student Perceptions and the Challenge of Correcting AI Errors. *International Journal of Artificial Intelligence in Education, 35,* 3166-3184. https://doi.org/10.1007/s40593-025-00496-4

Liao, J., Zhong, L., Zhe, L., Xu, H., Liu, M., & Xie, T. (2024). Scaffolding Computational Thinking with ChatGPT. *IEEE Transactions on Learning Technologies, 17,* 1628-1642. https://doi.org/10.1109/tlt.2024.3392896

Liu, M., Zhang, L. J., & Biebricher, C. (2024). Investigating Students' Cognitive Processes in Generative AI-Assisted Digital Multimodal Composing and Traditional Writing. *Computers & Education, 211,* Article 104977. https://doi.org/10.1016/j.compedu.2023.104977

Lodi, M., & Martini, S. (2021). Computational Thinking, between Papert and Wing. *Science & Education, 30,* 883-908. https://doi.org/10.1007/s11191-021-00202-5

Marcionetti, J., & Castelli, L. (2023). The Job and Life Satisfaction of Teachers: A Social Cognitive Model Integrating Teachers' Burnout, Self-Efficacy, Dispositional Optimism, and Social Support. *International Journal for Educational and Vocational Guidance, 23,* 441-463. https://doi.org/10.1007/s10775-021-09516-w

Massaty, M. H., Fahrurozi, S. K., & Budiyanto, C. W. (2024). The Role of AI in Fostering Computational Thinking and Self-Efficacy in Educational Settings: A Systematic Review. *IJIE (Indonesian Journal of Informatics Education), 8,* 49-61. https://doi.org/10.20961/ijie.v8i1.89596

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, Inc.

Pinski, M., & Benlian, A. (2023). AI Literacy—Towards Measuring Human Competency in Artificial Intelligence. *Proceedings of the Annual Hawaii International Conference on System Sciences* (pp. 165-174). Hawaii International Conference on System Sciences. https://doi.org/10.24251/hicss.2023.021

Ponzini, D., Adorni, G., Delzanno, G., & Guerrini, G. (2024). Toward the Use of Generative AI to Develop Computational Thinking by Supporting Problem Decomposition. *Ital-IA 2024: 4th National Conference on Artificial Intelligence*.

Rahman, M. M., & Watanobe, Y. (2023). ChatGPT for Education and Research: Opportunities, Threats, and Strategies. *Applied Sciences, 13,* Article 5783.

https://doi.org/10.3390/app13095783

Salomon, G., & Perkins, D. (2005). Do Technologies Make Us Smarter? Intellectual Amplification with, of, and through Technology. In R. J. Sternberg, & D. D. Preiss (Eds.) *Intelligence and Technology: The Impact of Tools on the Nature and Development of Human Abilities* (pp. 71-86). Lawrence Erlbaum Associates Publishers.

Sharma, K., Papavlasopoulou, S., & Giannakos, M. (2019). Coding Games and Robots to Enhance Computational Thinking: How Collaboration and Engagement Moderate Children's Attitudes? *International Journal of Child-Computer Interaction, 21,* 65-76. https://doi.org/10.1016/j.ijcci.2019.04.004

Sun, D., Boudouaia, A., Zhu, C., & Li, Y. (2024). Would Chatgpt-Facilitated Programming Mode Impact College Students' Programming Behaviors, Performances, and Perceptions? an Empirical Study. *International Journal of Educational Technology in Higher Education, 21,* 1-22. https://doi.org/10.1186/s41239-024-00446-5

Sun, L., Hu, L., Yang, W., Zhou, D., & Wang, X. (2020). STEM Learning Attitude Predicts Computational Thinking Skills among Primary School Students. *Journal of Computer Assisted Learning, 37,* 346-358. https://doi.org/10.1111/jcal.12493

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: 2304.11938.

Tikva, C., & Tambouris, E. (2021). Mapping Computational Thinking through Programming in K-12 Education: A Conceptual Model Based on a Systematic Literature Review. *Computers & Education, 162,* Article 104083. https://doi.org/10.1016/j.compedu.2020.104083

Tlili, A., Burgos, D., & Looi, C.-K. (2023). Guest Editorial: Creating Computational Thinkers for the Artificial Intelligence Eracatalyzing the Process through Educational Technology. *Educational Technology & Society, 26,* 94-98.

Usher, M. (2025). Generative AI vs. Instructor Vs. Peer Assessments: A Comparison of Grading and Feedback in Higher Education. *Assessment & Evaluation in Higher Education, 50,* 912-927. https://doi.org/10.1080/02602938.2025.2487495

Vuorikari, R., Kluzer, S., & Punie, Y. (2022). *Digcomp 2.2, the Digital Competence Framework for Citizens: With New Examples of Knowledge, Skills and Attitudes.* Publications Office European Union.

Walter, Y. (2024). Embracing the Future of Artificial Intelligence in the Classroom: The Relevance of AI Literacy, Prompt Engineering, and Critical Thinking in Modern Education. *International Journal of Educational Technology in Higher Education, 21,* Article No. 15. https://doi.org/10.1186/s41239-024-00448-3

Wang, T., Zhou, N., & Chen, Z. (2024). *Enhancing Computer Programming Education with LLMS: A Study on Effective Prompt Engineering for Python Code Generation.* arXiv:2407.05437.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational Thinking for All: Pedagogical Approaches to Embedding 21st Century Problem Solving in K-12 Classrooms. *TechTrends, 60,* 565-568. https://doi.org/10.1007/s11528-016-0087-7

Yan, Y., Chen, C., Hu, Y., & Ye, X. (2025). LLM-Based Collaborative Programming: Impact on Students' Computational Thinking and Self-Efficacy. *Humanities and Social Sciences Communications, 12,* 1-12. https://doi.org/10.1057/s41599-025-04471-1

Yang, Y., Tseng, C. C., & Lai, S. (2024). Enhancing Teachers' Self-Efficacy Beliefs in Ai-Based Technology Integration into English Speaking Teaching through a Professional Development Program. *Teaching and Teacher Education, 144,* Article 104582. https://doi.org/10.1016/j.tate.2024.104582

Yilmaz, R., & Karaoglan Yilmaz, F. G. (2023a). Augmented Intelligence in Programming Learning: Examining Student Views on the Use of ChatGPT for Programming Learning. *Computers in Human Behavior: Artificial Humans, 1,* Article 100005. https://doi.org/10.1016/j.chbah.2023.100005

Yilmaz, R., & Karaoglan Yilmaz, F. G. (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,* Article 100147. https://doi.org/10.1016/j.caeai.2023.100147

Zhang, L., KaLyuga, S., Lee, C., & Lei, C. (2016) Effectiveness of Collaborative Learning of Computer Programming under Different Learning Group Formations According to Students' Prior Knowledge: A Cognitive Load Perspective. *Journal of Interactive Learning Research, 2,* 171-192

Zhang, B., Liang, P., Zhou, X., Ahmad, A., & Waseem, M. (2023). Demystifying Practices, Challenges and Expected Features of Using Github Copilot. *International Journal of Software Engineering and Knowledge Engineering, 33,* 1653-1672. https://doi.org/10.1142/s0218194023410048

Zhao, G., Yang, L., Hu, B., & Wang, J. (2025). A Generative Artificial Intelligence (ai)-Based Human-Computer Collaborative Programming Learning Method to Improve Computational Thinking, Learning Attitudes, and Learning Achievement. *Journal of Educational Computing Research, 63,* 1059-1087. https://doi.org/10.1177/07356331251336154

Zheng, L., Zhen, Y., Niu, J., & Zhong, L. (2022). An Exploratory Study on Fade-In versus Fade-Out Scaffolding for Novice Programmers in Online Collaborative Programming Settings. *Journal of Computing in Higher Education, 34,* 489-516. https://doi.org/10.1007/s12528-021-09307-w