ASCILITE 2026

The Future is Connection:

Collaboration, Creativity and Compassion in a Digital Society

GenAI and novice programmers' conceptual understanding: A systematic review

Sonam Wangmo

James Cook University

Programming education remains challenging for novice learners, and the emergence of Generative Artificial Intelligence (GenAI) has introduced new opportunities and challenges for learning programming. While existing research reports a range of potential benefits, the influence of GenAI on conceptual understanding remains underexplored. This systematic literature review investigates the role of GenAI in influencing novice programmers' conceptual understanding in higher education programming contexts. Guided by the PRISMA framework, empirical studies published between 2022 and 2026 were reviewed and synthesised using a conceptual understanding framework comprising three dimensions: understanding programming constructs, reasoning about program behaviour, and developing mental models of program execution. The findings indicate that GenAI primarily functions as a learning support tool that may enhance understanding of programming constructs. However, evidence is less conclusive regarding reasoning about program behaviour and is extremely limited in relation to mental models of program execution. The review suggests that the effects of GenAI are concentrated in more surface-level dimensions of understanding and do not consistently translate into deeper conceptual understanding. The study identifies important gaps in current research and outlines directions for further investigation.

Keywords: Generative Artificial Intelligence (GenAI), programming education, conceptual understanding, novice programmer, higher education, systematic literature review

Introduction

Despite decades of research and pedagogical innovation, introductory programming remains a challenging area of computing education, characterised by high failure rates and persistent difficulties in developing and applying conceptual understanding (Lister et al., 2004; Robins et al., 2003). Accordingly, researchers have investigated how learners develop programming knowledge, encompassing understanding programming constructs, reasoning about program behaviour, and developing mental models of program execution (Lister et al., 2004; Robins et al., 2003; Sorva, 2013). Collectively, these dimensions provide a lens for understanding programming cognition and form the Conceptual Understanding (CU) framework adopted in this review, as presented in Table 1. More recently, the emergence of Generative Artificial Intelligence (GenAI) has transformed higher education. GenAI tools such as ChatGPT, Gemini, Claude, and others can generate code, explain programming concepts, provide examples, and assist with debugging, thereby significantly lowering the barriers to completing programming tasks (Chen et al., 2025; Haindl & Weinberger, 2024; Scholl & Kiesler, 2024).

Table 1

Dimensions of Conceptual Understanding

Dimension	Indicators
CU1 - Understanding programming constructs	Understanding programming concepts such as variables, decision structures, repetitions, functions, data structures, and their purpose and use (Robins et al., 2003).
CU2 - Reasoning about program behaviour	Code tracing, predicting outputs, explaining program execution, understanding why a program produces a particular result (Lister et al., 2004).

CU3 – Developing mental models of program execution	Understanding of the underlying execution process, including how variables, assignment, memory, and state changes interact during program execution (Sorva, 2013).

While these capabilities present new opportunities for learning, they also raise important questions regarding CU. It remains unclear whether the ability to generate functioning code with GenAI contributes to the development of such understanding among novice programmers. Although GenAI is increasingly studied in programming education, there is limited synthesis of how its use influences students' grasp of underlying concepts. The studies included in this review have reported a range of outcomes, including code understanding, syntax comprehension, debugging support, programming performance, and student experiences; however, these findings have rarely been evaluated collectively through a conceptual lens. To address this gap, this systematic review investigates the role of GenAI in influencing CU in higher education programming contexts. It is guided by the research question: What role does GenAI play in influencing novice programmers' conceptual understanding in higher education programming education? Drawing on 16 empirical studies, the review applies the CU framework presented in Table 1 to examine how GenAI shapes different dimensions of this understanding and to identify areas that remain underexplored.

Methodology

This research was conducted using the Systematic Literature Review (SLR) methodology guided by the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) framework (Page et al., 2021). To capture the recent developments in the use of GenAI in programming education, literature published between 2022 and 2026 was retrieved from three databases: IEEE Xplore (n = 94), Scopus (n = 31), and Google Scholar (n = 63, most relevant records screened), resulting in a total of 188 records. Database-specific search strategies were used, including keywords such as GenAI, introductory programming, novice programmers, conceptual understanding, and higher education, as shown for Scopus below:

> TITLE-ABS-KEY ("ChatGPT" OR "generative AI" OR "large language model*" OR "GitHub Copilot") AND
> TITLE-ABS-KEY ("introductory programming" OR "intro programming" OR "CS1" OR "novice programmers" OR beginners) AND
> TITLE-ABS-KEY ("higher education" OR universit*) AND
> TITLE-ABS-KEY ("conceptual understanding" OR understanding OR "problem solving" OR debugging)

Following the search process, duplicate records (n = 11) were removed, and the remaining 177 studies were then screened according to the inclusion and exclusion criteria outlined in Table 2, leaving 45 articles for full-text review. Finally, 16 empirical studies met the eligibility criteria and were included for the final synthesis.

Table 2
Inclusion and Exclusion Criteria

Inclusion Criteria	**Exclusion Criteria**
1. Includes GenAI in programming education	1. Not related to programming education or GenAI
2. Conducted in higher education settings	2. Focus on technical development of AI models without an educational or learning context
3. CS1, novice programmers, programming students	3. Did not involve higher education learners
4. Reported outcomes related to programming learning and understanding	4. Did not report learning-related outcomes
5. Peer-reviewed journal and conference papers in English	5. Books, theses, editorials, non-peer-reviewed works
6. Published between 2022–2026	6. Studies published outside the selected timeframe

Findings and Discussion

The findings from 16 empirical studies were synthesised using the CU framework presented in Table 1. The evidence relating to CU was grouped into three dimensions: understanding programming constructs (CU1), reasoning about program behaviour (CU2), and developing mental models of program execution (CU3). Figure 1 presents the distribution of direct (studies that explicitly examined outcomes related to programming

concepts, reasoning, or understanding) and indirect (studies that investigated factors such as learning behaviours, usage patterns, perceptions, reflective thinking, and programming performance that may influence conceptual understanding without directly measuring it) contributions across these dimensions.

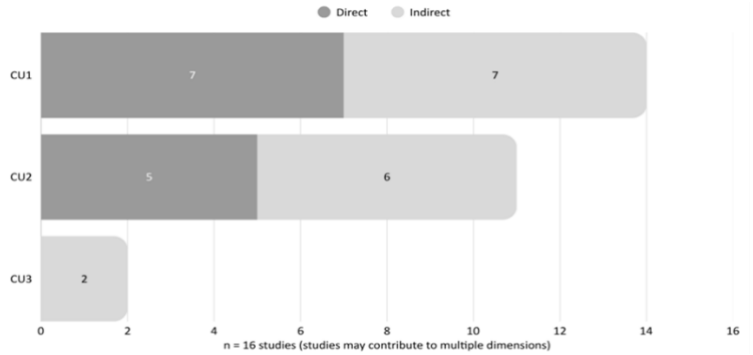


Figure 1. Distribution of contribution frequency across conceptual understanding dimensions

CU1 – Understanding Programming Constructs

As shown in Figure 1, CU1 was the most frequently represented dimension across the reviewed literature (n = 14), with equal numbers of direct (n = 7) and indirect (n = 7) contributions. Across the four studies included in this review, evidence suggests that GenAI may support learning activities associated with understanding programming constructs through syntax assistance, code examples, concept explanations, and coding support (Chao & Chang, 2025; Haindl & Weinberger, 2024; Suh et al., 2025; Tadimalla et al., 2025). These studies observed positive outcomes relating to syntax comprehension, coding practices, and reinforcement of programming concepts. In contrast, the following three studies reported limited or mixed evidence of improved learning gains, with findings indicating that GenAI-assisted learners did not consistently demonstrate stronger understanding of programming concepts or learning outcomes compared with learners using alternative approaches or relying less on GenAI support (Chen et al., 2025; Cubillos et al., 2025; Xue et al., 2024). Similarly, four studies found that improvements in productivity, task completion, or programming performance did not necessarily correspond to stronger learning outcomes (Cubillos et al., 2025; Das et al., 2025; Gomes et al., 2025; Mellado et al., 2024). Together, these findings suggest that although GenAI may support programming-related learning activities and performance, improvements in productivity and task completion should not be assumed to reflect stronger conceptual understanding of programming constructs.

CU2 – Reasoning About Program Behaviour

CU2 was represented in 11 studies, comprising direct (n = 5) and indirect (n = 6) contributions, as shown in Figure 1. Evidence relating to reasoning about program behaviour was limited and less consistent across the reviewed studies. For example, four studies found that GenAI-assisted learners did not consistently demonstrate stronger reasoning, understanding of solutions, or learning outcomes compared with learners using alternative approaches or relying less on AI support (Chao & Chang, 2025; Chen et al., 2025; Cubillos et al., 2025; Xue et al., 2024). Beyond learning outcomes, three studies highlighted the influence of GenAI on learners' reasoning processes. Park et al. (2025) suggested that dependency on GenAI may limit the development of independent problem-solving abilities and long-term skill retention. Similarly, Gomes et al. (2025) identified that students expressed concerns that reliance on ChatGPT could hinder reasoning development and learning. On the other hand, Rachmat et al. (2025) found that GenAI-supported interactions may encourage reflective thinking by prompting learners to justify and evaluate their understanding. Collectively, these findings suggest that although GenAI may shape how learners approach reasoning-related programming tasks, its contribution to strengthening reasoning about program behaviour remains unclear.

CU3 – Developing Mental Models of Program Execution

CU3 was the least represented dimension across the reviewed literature (n = 2), with no direct contributions and only two indirect contributions displayed in Figure 1. None of the reviewed studies explicitly examined how GenAI influences learners' mental models of program execution, including their understanding of

variables, assignment, memory, and state changes during program execution. Chen et al. (2025) and Park et al. (2025) discussed learners' understanding of solutions, underlying concepts, and independent problem-solving abilities, which may be associated with the development of mental representations of program execution. However, neither study directly investigated learners' mental models, making the relationship inferential rather than explicit. Consequently, the role of GenAI in supporting learners' mental models of program execution remains largely unexplored within the current literature.

Across the three dimensions of CU, learner interaction with GenAI emerged as a recurring factor influencing educational outcomes. Seven studies examined how students used GenAI during programming activities and observed considerable variation in usage patterns (Amoozadeh et al., 2024; Dillon et al., 2025; Gomes et al., 2025; Rachmat et al., 2025; Scholl et al., 2024; Scholl & Kiesler, 2024; Tadimalla et al., 2025). Students used GenAI for a range of purposes, including generating code, obtaining programming support, debugging, solving problems, answering questions, and engaging in iterative interactions. These findings suggest that the educational impact of GenAI may depend not only on the technology itself but also on how learners engage with it. Differences in usage patterns may partly explain the mixed findings shown across the CU1, CU2, and CU3 dimensions.

Future Directions and Conclusion

Viewed through the CU1–CU2–CU3 framework, this review found that GenAI can support novice programmers through code generation, explanations, examples, and programming assistance. However, evidence that these benefits translate into stronger conceptual understanding remains limited. The strongest body of evidence relates to CU1, while findings for CU2 are less conclusive and evidence relating to CU3 is extremely limited. Moreover, improvements in programming performance or task completion do not consistently correspond to deeper understanding. Collectively, these findings suggest that the benefits of GenAI are concentrated in more surface-level dimensions of understanding and diminish as cognitive demands increase.

These findings have important implications for programming education, as performance and productivity may improve without a corresponding gain in conceptual understanding. Such measures may therefore be unreliable proxies for learning, with learners appearing successful on programming tasks while their underlying understanding remains unclear. For educators and curriculum designers, this raises the risk of conflating productivity with understanding. The scarcity of evidence addressing CU3 is especially consequential, as mental models of program execution support debugging, prediction, and the transfer of knowledge to new problems; this gap therefore lies at a foundational layer of programming competence. The inconclusive evidence for CU2 is similarly important, as reasoning about program behaviour distinguishes genuine understanding from simple pattern reproduction.

Future research should directly examine how GenAI influences reasoning about program behaviour and the development of mental models of program execution, while moving beyond programming performance measures to assess conceptual understanding more explicitly. This will require measures that isolate conceptual understanding such as explanation, prediction, and debugging tasks, rather than relying on completion or productivity outcomes. Given that many existing studies involve small samples, specific educational contexts and short intervention periods, broader and longitudinal investigations are needed to better understand the long-term impact of GenAI on programming learning. As GenAI becomes increasingly integrated into programming education, understanding how it can support meaningful learning rather than simply task completion remains an important area for future research. Addressing these questions will provide a clearer understanding of whether GenAI strengthens or erodes the conceptual foundations of programming.

References

Amoozadeh, M., Nam, D., Prol, D., Alfageeh, A., Prather, J., Hilton, M., Ragavan, S. S., & Alipour, A. (2024). Student-AI interaction: A case study of CS1 students. In *Proceedings of the 24th Koli calling international conference on computing education research (Koli Calling '24)* (Article 13, pp. 1–13). Association for Computing Machinery. https://doi.org/10.1145/3699538.3699567

Chao, H.-W., & Chang, A. Y. (2025). Exploring the role of AI in transforming programming education at universities: Qualitative and quantitative insights. In *2025 International Workshop on Artificial Intelligence and Education (WAIE)* (pp. 173–180). IEEE. https://doi.org/10.1109/WAIE67422.2025.11381160

Chen, R., Jiang, S., Shen, J., Moon, A. J., & Wei, L. (2025). Examining the usage of generative AI models in student learning activities for software programming. In *2025 2nd IEEE/ACM International Conference on AI-Powered Software (AIware)* (pp. 76–86). IEEE. https://doi.org/10.1109/AIware69974.2025.00016

Cubillos, C., Mellado, R., Cabrera-Paniagua, D., & Urra, E. (2025). Generative artificial intelligence in computer programming: Does it enhance learning, motivation, and the learning environment? *IEEE Access*, *13*, 40438–40455. https://doi.org/10.1109/ACCESS.2025.3532883

Das, U. K., Yates, M. S., Bloch, K. A., Hagan, L., Thomas, E. L., Gatos, G., Rajkumar, A. K., Valavala, B., Franco, S., Divvela, S. K., Thompson, A. K., & Bego, C. R. (2025). The impact of a GenAI integration on first-year engineering students' programming outcomes. In *2025 IEEE Frontiers in Education Conference (FIE)* (pp. 1–6). IEEE. https://doi.org/10.1109/FIE63693.2025.11328708

Dillon, E., Ordóñez, P., Hasan, U., Owoeye, O. N., & Dharmale, S. R. (2025). Exploring the impact of ChatGPT on early information systems majors: Opportunities and challenges in learning to program. In *2025 IEEE Frontiers in Education Conference (FIE)* (pp. 1–9). IEEE. https://doi.org/10.1109/FIE63693.2025.11328626

Gomes, L. N., Fraga, C. F., Holanda, M., & Da Silva, D. (2025). ChatGPT as a teaching assistant in a CS1 course: An experience report. In *2025 IEEE Frontiers in Education Conference (FIE)* (pp. 1–9). IEEE. https://doi.org/10.1109/FIE63693.2025.11328432

Haindl, P., & Weinberger, A. G. (2024). Does ChatGPT help novice programmers write better code? Results from static code analysis. *IEEE Access*, *12*, 114146–114156. https://doi.org/10.1109/ACCESS.2024.3445432

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J. E., Sanders, K., Seppälä, O., Simon, B., & Thomas, L. (2004). A multi-national study of reading and tracing skills in novice programmers. *ACM SIGCSE Bulletin*, *36*(4), 119–150. https://doi.org/10.1145/1041624.1041673

Mellado, R., Cubillos, C., & Ahumada, G. (2024). Effectiveness of generative artificial intelligence in learning programming to higher education students. In *2024 IEEE International Conference on Automation/XXVI Congress of the Chilean Association of Automatic Control (ICA-ACCA)* (pp. 1–7). IEEE. https://doi.org/10.1109/ICA-ACCA62622.2024.10766746

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., … Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, *372*, n71. https://doi.org/10.1136/bmj.n71

Park, S. M., Ho, M., Lin, M. P.-C., & Ryoo, J. (2025). Evaluating the impact of assistive AI tools on learning outcomes and ethical considerations in programming education. In *2025 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1–10). IEEE. https://doi.org/10.1109/EDUCON62633.2025.11016517

Rachmat, A., Watterson, C., & Lundqvist, K. (2025). The impact of chatbots on students' reflective thinking in introductory programming course. In *2025 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1–10). IEEE. https://doi.org/10.1109/EDUCON62633.2025.11016417

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Scholl, A., & Kiesler, N. (2024). How novice programmers use and experience ChatGPT when solving programming exercises in an introductory course. In *2024 IEEE Frontiers in Education Conference (FIE)* (pp. 1–9). IEEE. https://doi.org/10.1109/FIE61694.2024.10893442

Scholl, A., Schiffner, D., & Kiesler, N. (2024). Analyzing chat protocols of novice programmers solving introductory programming tasks with ChatGPT. In S. Schulz & N. Kiesler (Eds.), *Proceedings of DELFI 2024* (pp. 63–79). Gesellschaft für Informatik e.V. https://doi.org/10.18420/delfi2024_05

Sorva, J. (2013). Notional machines and introductory programming education. *ACM Transactions on Computing Education*, *13*(2), Article 8. https://doi.org/10.1145/2483710.2483713

Suh, J., Lee, K., & Lee, J. (2025). Programming education with ChatGPT: Outcomes for beginners and intermediate students. *Education and Information Technologies*, *30*(14), 19511–19536. https://doi.org/10.1007/s10639-025-13542-4

Tadimalla, S. Y., Maher, M. L., Najjar, N., & Simon, C. (2025). The impact of AI tools on the student programming experience: Interactions, attitudes & perceptions. In *2025 IEEE Frontiers in Education Conference (FIE)* (pp. 1–10). IEEE. https://doi.org/10.1109/FIE63693.2025.11328137

ASCILITE 2026

The Future is Connection:

Collaboration, Creativity and Compassion in a Digital Society

Xue, Y., Chen, H., Bai, G. R., Tairas, R., & Huang, Y. (2024). Does ChatGPT help with introductory programming? An experiment of students using ChatGPT in CS1. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '24)* (pp. 331–341). Association for Computing Machinery. https://doi.org/10.1145/3639474.3640076

Wangmo, S. (2026). GenAI and novice programmers' conceptual understanding: A systematic review. In Cochrane, T., Narayan, V., Poole, T., Lai, P., Lake, P., Moore, O., & Li, L. (Eds.), *The Future is Connection: Collaboration, Creativity and Compassion in a Digital Society*. Proceedings ASCILITE 2026. Brisbane (pp. 877–882). https://doi.org/10.65106/apubs.2026.3134

Note: All published papers are refereed, having undergone a double-blind peer-review process.
The author(s) assign a Creative Commons by attribution license enabling others to distribute, remix, tweak, and build upon their work, even commercially, as long as credit is given to the author(s) for the original creation.

© Wangmo, S. 2026