Annales Mathematicae et Informaticae
61 (2025) pp. 141–155
DOI: 10.33039/ami.2025.10.020
URL: https://ami.uni-eszterhazy.hu

Algorithmic thinking at risk? Exploring LLM use in computer science education

Sándor Király, Ede Troll

Eszterházy Károly Catholic University
{kiraly.sandor,troll.ede}@uni-eszterhazy.hu

Abstract. The rapid rise of AI – especially Large Language Models (LLMs) like GPT-4, Microsoft Copilot, and Google Gemini – has significantly impacted higher education. LLMs support students in problem-solving, writing, and learning complex topics, while educators use them for course planning, lecture content, and assessments. The primary aim of this research was to explore whether university computer science students use large language models (LLMs) to support their learning, and if so, how and why. The study was conducted among students enrolled in a three-year BSc in Computer Science program at Eszterházy Károly Catholic University. The study combined questionnaires with semi-structured interviews involving nine students and three instructors. Students reported using AI chatbots for tasks such as code testing, debugging, understanding examples, generating code, designing exercises, and self-assessment. LLM usage increased with subject complexity and varied by programming skill. While students were moderately satisfied with LLMs, instructors voiced concerns that overreliance could undermine algorithmic thinking and coding skills. The findings suggest a need to revise assessment methods and enhance teaching materials to better reflect current educational practices.

Keywords: computer science education, LLMs, AI Chatbots

AMS Subject Classification: 97D40, 68Q70

1. Introduction

The rapid advancement of artificial intelligence (AI), particularly Large Language Models (LLMs) such as OpenAI's GPT-4, Microsoft's CoPilot, and Google's Gem-

Accepted: October 19, 2025
Published online: October 28, 2025

ini, has significantly impacted higher education. LLMs – AI models trained on massive text datasets – can generate and understand human-like language, enabling applications beyond chatbots, including summarization, translation, code generation, and question answering. Chatbots are just one interface that leverage LLMs, often enhanced with tools like memory, personality, or calculators.

In education, LLMs assist students with problem-solving, writing, and concept explanation ([1, 7, 10, 33], while educators use them for course planning, content generation, and assessment design [6, 27]. In computer science education, they offer code generation, debugging, and natural language explanations [8, 9, 29, 34, 36], presenting both opportunities and challenges. Despite their benefits in content creation and real-time support, issues like contextual misinterpretation, bias, and ethical concerns persist [2, 7, 28].

Programming skills develop through practice, and deep-learning tools now support tasks like code repair, completion, and verification [23–26, 37]. Transformer-based LLMs like CodeBERT, Codex, and PyMT5 have achieved state-of-the-art results [32]. ChatGPT, based on this architecture, is widely used for its human-like interaction style [23]. These tools help students navigate programming challenges by offering debugging and problem-solving assistance [3, 11, 18, 19, 35, 36].

Some studies report improved learning outcomes with LLM use – for instance, Akçapınar and Sidan observed higher exam scores when students used a custom AI assistant [3]. However, they also noted a tendency to accept incorrect outputs uncritically, highlighting the importance of careful integration. Additional concerns include plagiarism, academic dishonesty, and content reliability [13].

Given these opportunities and risks, understanding how and why students use LLMs is vital for developing effective teaching practices. This study explores computer science students' motivations and usage patterns with LLMs, particularly in foundational programming courses, and examines possible links to academic performance.

The research questions guiding this study are

(RQ1): How do computer science students use language models in their learning process?

(RQ2): Do all students use language models in programming in the same ways and for the same purposes?

2. Literature review

Recent research highlights both the benefits and limitations of LLM-based assistants in programming education. Ravšelj et al. [28] found that students across 109 countries primarily used ChatGPT for brainstorming and summarizing, reporting generally positive attitudes. Similarly, Alves et al. reviewed studies on AI chatbots in programming and found positive impacts on learning, but also noted gaps, such as limited focus on teachers' views and student collaboration [4].

Experimental studies have shown improved performance with AI support [15, 16, 21], especially in coding tasks, debugging, and personalized learning. However,

Groothuijsen et al. [12] and Xue et al. [35] reported concerns about reduced collaboration, learning outcomes, and motivation when ChatGPT was used. Students often used AI not to copy code directly, but to structure or debug their work.

Several studies point to the unreliability of AI-generated code. Liu et al. [18] found that many ChatGPT solutions, though functional, suffered from poor maintainability. Chu et al. [5] and Rahman & Watanobe [24] noted issues such as hallucinations, bias, and lack of reasoning. Akçapınar & Sidan [3] observed a 54% improvement in exam scores with AI use, but also that most students accepted incorrect answers uncritically.

While students appreciate the interactivity and support offered by AI tools [19], overreliance may hinder independent learning. Most studies emphasize AI's positive role [1, 11, 31, 36], though some warn of negative effects on academic integrity and engagement [14, 17, 22].

At our institution, a steady decline in programming course grades has been observed despite stable assessments. This trend may reflect reduced student engagement with algorithmic thinking, possibly linked to increased use of AI-generated code. To explore this, our study investigates how and why students use AI tools and whether their usage relates to skill development.

3. Methods

3.1. Participants

The study was conducted in December 2024 among students enrolled in a three-year Bachelor of Science program in Computer Science at a Hungarian university, encompassing both full-time and part-time cohorts. Two instruments – Questionnaire A and Questionnaire B – were administered to a total of 256 students across all three academic years. Questionnaire A was completed by 232 students, while 211 students responded to Questionnaire B. Participation in both surveys was entirely voluntary and anonymous.

Additionally, semi-structured interviews were carried out with nine students representing different year groups and three instructors who were not affiliated with the authorship of this study. Participants were selected through random sampling, with deliberate inclusion of individuals exhibiting low, average, and high academic performance across each year group. Instructors were likewise randomly selected from among those teaching programming-related subjects.

3.2. Data collection

Data for this mixed-methods study were collected using four approaches. First, Questionnaire A was administered to 256 students at semester's end, using a five-point Likert scale to assess their use of large language models (LLMs). It covered model types, usage contexts and frequency, trust, familiarity with underlying concepts, and demographic details. The primary aim was to determine when and how

students engaged with LLMs and the extent of their trust in these tools.

Second, Questionnaire B, also distributed to the same cohort, had two parts. B1 employed the validated Technology Readiness Index 2.0 (TRI 2.0; [20]), measuring Optimism, Innovativeness, Discomfort, and Insecurity toward new technologies. B2, developed for this study, collected students' self-reported grades in three progressively advanced programming courses (HLPL1, HLPL2, SOP) and examined their use of LLMs during coursework and assessments.

Third, semi-structured group interviews were conducted with nine student volunteers (three per year group) to explore their LLM use, engagement, and perceptions of skill development.

Fourth, three instructors were interviewed to gather perspectives on LLM integration in teaching, student engagement, skill development, and anticipated curricular changes. Combining quantitative and qualitative data from students and instructors enabled source and method triangulation. The validated TRI 2.0 instrument and rigorously developed interview protocols enhanced reliability.

3.3. Data analysis

For the TRI 2.0 section of the questionnaire, mean scores for each of the four subscales – Optimism, Innovativeness, Discomfort, and Insecurity – were calculated using SPSS, with each subscale comprising four items. Cronbach's alpha indicated acceptable internal consistency for Optimism ($\alpha = .76$) and Innovativeness ($\alpha = .76$), while Discomfort ($\alpha = .60$) and Insecurity ($\alpha = .54$) showed lower reliability, warranting cautious interpretation and suggesting a need for potential refinement in future applications.

To address Research Question 2 (RQ2), several statistical analyses were conducted in SPSS. In addition, qualitative analysis was performed on the open-ended questionnaire responses and the student group interview notes, following Braun and Clarke's (2006) six-phase thematic analysis. Sensitizing concepts from Rahman and Watanobe [23] – error checking and debugging, conceptual understanding support, code generation, and code optimization – guided the initial coding process.

Semi-structured interviews with nine students and three instructors involved in the HLPL1, HLPL2, and SOP courses were also analysed thematically. The interview protocol was designed to explore participants' experiences, motivations, and concerns regarding AI chatbot use in programming education. Interviews lasted 30–45 minutes, were audio-recorded, transcribed verbatim, and analysed manually to enable cross-case comparisons.

Coding included both inductive and deductive approaches. Emergent codes were grouped into candidate themes, which were then refined for clarity and consistency. Themes were compared across courses and between student and teacher perspectives, with representative quotes selected to illustrate key findings.

This process facilitated a nuanced understanding of how and why AI chatbots are used in programming education, highlighting implications for student autonomy, skill development, and assessment practices. A similar approach was applied

to the teacher interview data, synthesizing their insights on student engagement, learning outcomes, and anticipated curricular adjustments.

All procedures for data collection, storage, and analysis adhered to the ethical standards set by the university's Research Ethical Review Board and complied fully with the General Data Protection Regulation (GDPR, 2016).

4. Results

4.1. Descriptive results

Of the 256 students invited, 232 completed the first questionnaire. Among respondents, 33.6% were first-year, 33.2% second-year, and the remainder third-year students. The sample comprised 15% female and 75% male participants. Among the respondents, 96.6% had heard of ChatGPT, 33.2% knew about Claude, and 30.8% knew about LLaMA. 1% of respondents had never used a chatbot, while 17.9% used them very frequently. On a Likert scale from 1 to 5, the average chatbot usage frequency was 3.44.

4.1.1. Research Question 1: How do computer science students use large language models in their learning process?

The majority of students reported using LLMs, indicating a higher likelihood of engagement than non-use. The average value (Likert scale 1-5) is 3.45. Respondents rarely use chatbots for text translation (2.39), still preferring traditional applications like Google Translate and DeepL (see Figure 1).

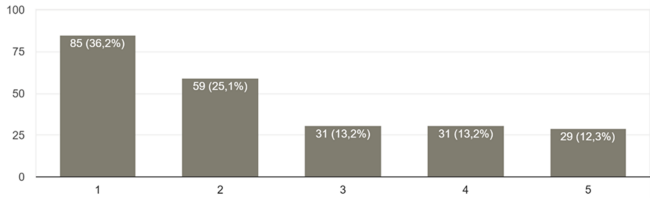

Figure 1. How often do you use a large language model for text translation (more often than e.g. Google Translate, DeepL, etc.)?

They also rarely use chatbots for non-academic conversations, with an average response of 2.83. When asked how often they use AI chatbots to solve homework assignments, the average response was 3.46. Disaggregated by year, first-year students reported an average of 3.29, while third-year students averaged 3.74. These results show that language models are mainly used for homework, not for translation or conversation.

Meanwhile, 64% of respondents either did not use AI at all or used it very rarely at the beginning of their programming studies.

Based on the questionnaire responses, it is evident that students prefer using AI tools for code generation. Of the 232 respondents, only 19 (8%) reported never having generated code using an AI tool. The overall average frequency for code generation was 3.11 on a five-point scale, with first-year students averaging 2.67 and third-year students 3.6 (Table 1). Asked how often they integrate AI-generated code into their programs without verification, the average response was 1.55. 59% had never done this, while 87% only used AI-generated code after verification.

Table 1. What did students use LLMs for?

	Never	Average		
Students used LLMs for:		First year	Second year	Third year
Code generation	8%	2.67	3.12	3.6
Code generation and submitted without verification	59%	1.42	1.5	1.7
Code generation and submitted with verification	13%	–	–	–
Debugging	12%	3.32	3.5	3.8

Table 1 shows that as students progress in their studies, they use LLM more and more often.

When learning a new programming language or technology, only 2.1% of students reported primarily relying on AI tools, while the majority preferred traditional resources such as YouTube, textbooks, and other learning materials (see Figure 2).

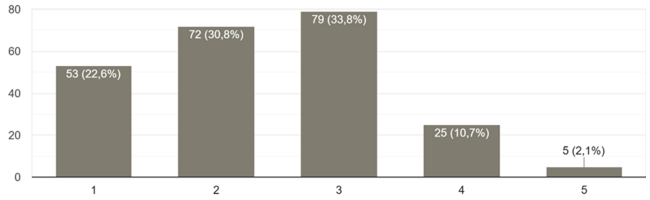

Figure 2. What do you rely on most when learning a new programming language or technology? (1–Youtube, Udemy, books, etc. 5–LLMs)

Questionnaire B

Responses to the first questionnaire offered an overview of LLM usage among IT students, highlighting both common applications and areas of non-use. The ques-

tionnaire also explored individual differences in usage patterns, purposes, and perceived usefulness of AI chatbots.

A total of 211 students completed the second questionnaire: 54 first-year, 63 second-year, and 99 third-year students. Questionnaire B incorporated the Technology Readiness Index (TRI 2.0). As shown in Table 2, Optimism and Innovativeness received the highest scores, while Discomfort was rated low and Insecurity moderate. These results indicate that participants are technologically adept and approach new technologies with a critically reflective mindset.

Table 2. Overview of TRI2 dimension scores.

Overview TRI2 scores		
Dimension	Mean	SD
Optimism	3.86	0.73
Innovativeness	3.50	0.81
Dissatisfaction	2.60	0.73
Insecurity	2.44	0.58

The results indicate that students exhibit both comfort with and openness toward adopting new technologies.

Table 3 shows how often students use AI chatbots to explain, review, test, and debug. Only 34 (16%) students reported never using AI for these purposes. Ninety students (43%) indicated usage between 1-10 times, 43 students (20%) between 10-20 times, and 44 (21%) students more than 20 times. Even in the Service-Oriented Programming (SOP) course – the course with the lowest reported usage – only 24% of students reported using AI for reviewing and testing their own code.

Figure 3 illustrates that students were generally moderately satisfied with the large language models (LLMs) they used, with an average satisfaction rating of 3.30. The relationship between satisfaction and students' course grades, as well as overall GPA, was also analysed. Students who received grades of 4, 5, or 1 reported marginally higher satisfaction levels (3.36, 3.39, and 3.37, respectively).

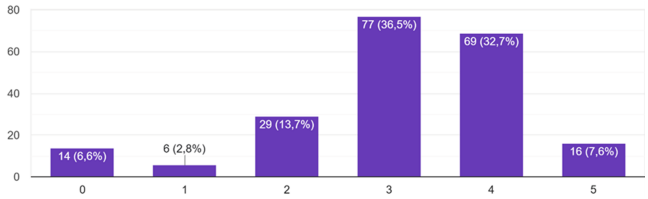

Figure 3. How satisfied are you with the language model(s) used and their responses? (0 - I do not use it, 1 - I am not satisfied, 5 - I am very satisfied)

Table 3. Student responses on AI use for explaining, reviewing, testing, and debugging.

Questions	Number of students
How often did students use AI for explaining, reviewing, testing, and debugging? – "Never" responses	34
How often did students use AI for explaining, reviewing, testing, and debugging? – 1–10 times	90
How often did students use AI for explaining, reviewing, testing, and debugging? – 10–20 times	43
How often did students use AI for explaining, reviewing, testing, and debugging? – More than 20 times	44

4.1.2. Research Question 2: Do all students use language models in programming in the same ways and for the same purposes?

After processing the Questionnaire B, we found that the method of LLM use depends on the subject and grades. We performed statistical analysis for the three subjects as follows. *Dependent Variable: Grade (A): Numerical grade (1–5). Independent Variables: AI Usage Variables: General AI frequency (B), AI-generated code (C), AI code not understood (D), AI code understood (E), AI-reviewed code (F), AI debugging (G).*

In the case of the HLPL1, the analysis demonstrated significant negative associations between AI usage and academic performance, particularly when students relied on AI-generated code without full understanding ($\rho = -0.35$, $p < 0.001$). The performed cluster analysis further identified a high-risk group (25% of students) with heavy AI dependence and markedly lower grades (M = 2.1). Regression models confirmed these effects persisted after accounting for baseline skill levels ($\beta = -0.42$, $p < 0.001$), underscoring the need for pedagogical strategies that promote critical engagement with AI chatbots (Table 4).

Similar results were found for the High-Level Programming Languages II subject. The statistical analysis for correlation shows that negative correlations between higher AI usage and lower grades. General AI usage (B), debugging (G) and lack of understanding (D) show the most significant negative relationships. This suggests that frequent, less reflective AI use may associate with poorer academic performance.

Using AI generated codes without understanding (C) and AI usage frequency ford debugging (G) significantly predict lower grades.

Table 4. Correlation between HLPL1 grades and AI chatbot usage using Spearman's method (with $p < 0.05$ for significance).

AI usage types	**Correlation (ρ)**	**p-value**	**Interpretation**
(B) How often did students use an LLM?	-0.25	< 0.001	Moderate negative link
(C) How often did students use AI-generated code to solve assignments? (Only tested if it worked, without modifications)	-0.32	< 0.001	Strong negative link
(D) How often did students use AI-generated code without understanding every line?	-0.35	< 0.001	Strongest negative effect
(E) How often did students use AI-generated code while fully understanding every line?	-0.10	0.132	No significant link
(F) How often did students use AI to review or test their own written code	-0.14	0.021	Weak negative link
(G) How often did students use AI for debugging?	-0.18	0.003	Moderate negative link

These findings underscore that AI's role in learning is context-dependent: while tool-assisted comprehension (e.g., debugging with understanding) may be neutral, unreflective dependence correlates with academic risks.

In the case of SOP, the results are different. The statistical analysis for correlation shows that higher-graded students used AI-generated code for assignments slightly less often ($r = -0.21$, $p = 0.022$). All other AI usage behaviours showed no meaningful relationship with grades (Table 5).

Unlike the other two courses, lower-performing students in this course also avoided using AI chatbots for code generation. Semi-structured group interviews revealed that large language models (LLMs) struggle with stream-based communication tasks, often producing non-functional or algorithmically inconsistent code. Instructors observed that LLM-generated solutions to producer-consumer problems were overly complex, included irrelevant code, and remained unclear even with explanations. As one teacher noted: *"AI is not yet sufficiently reliable for addressing more complex problems and should not be relied upon by individuals pursuing a serious career in this field."* Another added: *"For SOP, the situation is similar – obtaining a usable response may require an excessive number of prompts. The model may also struggle with concepts such as asynchronous programming."* Conse-

Table 5. Correlation between SOP grades and AI chatbot usage using Spearman's method (with $p < 0.05 =$ significant).

AI usage types	**Correlation (ρ)**	**p-value**	**Interpretation**
(B) How often did students use an LLM?	0.02	< 0.85	No significant correlation
(C) How often did students use AI-generated code to solve assignments? (Only tested if it worked, without modifications)	-0.21	< 0.022	Weak negative correlation
(D) How often did students use AI-generated code without understanding every line?	-0.11	< 0.24	No significant correlation
(E) How often did students use AI-generated code while fully understanding every line?	0.02	0.81	No significant correlation
(F) How often did students use AI to review or test their own written code	0.05	0.58	No significant correlation
(G) How often did students use AI for debugging?	0.07	0.44	No significant correlation

quently, students primarily used chatbots in this course to explain code rather than to generate or debug it. Across all three courses, a clear trend emerged: students with lower grades tended to rely on AI chatbots more frequently and were more likely to use them for code generation. In contrast, higher-achieving students used chatbots mainly for code explanation, testing, and debugging.

5. Discussion

This section addresses the study's findings in relation to the two research questions. The primary aim was to examine how university-level computer science students use large language models (LLMs) to support their learning and in what ways.

Survey results show that students primarily use OpenAI's ChatGPT, rarely for translation or casual conversation, but frequently for assignments, coursework, and exam preparation. As expected [12, 16, 23], they commonly use LLMs to generate, debug, analyse, and explain code. Only 8% reported never inserting LLM-generated code into their work, and just 59% claimed never to have submitted unverified AI-generated code – figures that suggest only a minority consistently

code independently (Table 1).

LLM usage increases with academic progression, likely due to both the growing reliability of LLMs and the increasing complexity of course content.

Some students used chatbots to generate exercises or quizzes for self-assessment, although specific course associations were not identified. This behavior aligns with RQ1, indicating diverse and evolving uses of LLMs in learning.

For RQ2, no prior studies were found that explore correlations between AI use and student ability. Our findings reveal that students struggling with a subject used LLMs more frequently – especially for code generation and less for code debugging. In contrast, higher-achieving students mainly used AI for testing and debugging. In HLPL1, over 90% did not use AI at all, while usage was notably higher in SOP. A positive correlation was found between perceived subject difficulty and AI use.

Students with lower grades often relied on LLMs to generate code, while stronger students used them for support tasks. Whether low performance resulted from excessive reliance on AI or pre-existing skill deficits remains unclear.

Regardless of performance, many students used AI to explain code. According to some, AI was unnecessary for SOP tasks as they could be solved using class material and logic. Others warned that excessive reliance on AI hindered genuine learning. Students expressed concern that AI-generated code is often syntactically correct but overly complex and difficult to understand, especially for weaker learners.

Instructors agreed that while LLMs can support learning, especially in code comprehension and debugging, generated code is often misaligned with course-specific approaches. Weaker students tend to depend on AI because they struggle to write or understand code. As tasks grow in complexity, AI-generated solutions may hinder more than help, particularly when students cannot replicate the solution independently.

Despite these concerns, students appreciated LLMs for saving time – especially when used to clarify errors or concepts they could not understand otherwise. Instructors cautioned, however, that using AI at the first sign of difficulty may impede long-term skill development.

Most students reported satisfaction with their chosen LLM (mean rating: 3.30), especially those who earned high or failing grades. They believed AI support enhanced their learning, reduced effort, and saved time.

As one part-time student observed, LLMs are useful in professional contexts – for generating SQL, HTML, or JSON – yet they tried to minimize reliance on AI in academic settings. Instructors echoed the need for caution, emphasizing that students who rely heavily on AI risk not learning essential problem-solving skills.

5.1. Implications

In subjects where students seek additional practice but find the number of available exercises insufficient, they often use AI to generate tasks. Instructors should address this by providing a wider range of high-quality exercises, including examples

with solutions and explanations, followed by similar tasks to reinforce understanding.

Self-assessment opportunities should be integrated into each course, enabling students to gauge their comprehension independently.

The findings also suggest that online exams may not reliably assess actual knowledge. As one respondent noted, *"more perceptive students may learn to obscure AI-generated syntax patterns, including stylistic features such as comments or distinctive variable naming conventions.*" Another added, "*every exam submission should be followed by a defense session, which would require additional time and effort from both instructors and students.*" To maintain academic integrity, exams should be conducted in person, under supervision, with mobile phone collected beforehand.

6. Conclusion and future work

This study found that computer science students use AI chatbots not only for code generation, explanation, testing, and debugging, but also for creating practice exercises and self-assessment, such as generating topic-specific test questions. Translation and non-academic use are rare. The primary motivation is to speed up learning and complete tasks more efficiently, though this is not always achieved. Code quality varies across chatbots, and faulty or overly complex outputs may hinder learning, even when explanations are provided.

Relying on AI-generated code can limit students' development of key skills such as algorithmic thinking and coding proficiency. Weaker students, who would benefit most from practice, are more likely to depend on code generation. Similarly, excessive use of AI tools for debugging may impede deeper code comprehension. Conversely, AI-generated explanations – particularly when applied to students' own code – can support learning when used appropriately.

The TRI 2.0 instrument showed acceptable reliability and partial construct and criterion validity in this context. Positive dimensions (optimism and innovativeness) performed well, though negative subscales may require refinement. Overall, the tool effectively measured students' technology readiness in relation to AI chatbot use.

Future studies could compare different AI tools and their respective affordances in education. Beyond code generation, AI can also evaluate student-written code [30], offering potential for both student self-assessment and instructional support. Investigating the impact of selecting appropriate AI tools and prompting strategies may further clarify their role in programming education.

References

[1] S. Abdulla, S. Ismail, Y. Fawzy, A. Elhaj: *Using ChatGPT in teaching computer programming and studying its impact on students' performance*, Electronic Journal of E-Learning 22.6 (2024), pp. 66–81, doi: 10.34190/ejel.22.6.3380.

[2] A. Acerbi, J. M. Stubbersfield: *Large language models show human-like content biases in transmission chain experiments*, PNAS 120.44 (2023), e2313790120, doi: 10.1073/pnas.2313790120.

[3] G. Akçapinar, E. Sidan: *AI chatbots in programming education: guiding success or encouraging plagiarism*, Discov Artif Intell 4 (2024), p. 87, doi: 10.1007/s44163-024-00203-7.

[4] J. V. B. Alves, Y. T. Gonçalves, H. B. Silva: *Use of ChatBots in Programming Education: A Scoping Review*, in: XIII Congresso Brasileiro de Informática na Educação (CBIE 2024), Rio de Janeiro, 2024, doi: 10.5753/sbie.2024.242473.

[5] Z. Chu, J. Wang, J. Xie, T. Zhu, Y. Yan, J. Ye, A. Zhong, X. Hz, J. Liang, P. S. Yu, O. Wen: *LLM Agents for Education: Advances and Applications*, 2024, doi: 10.48550/arXiv.2503.11733, eprint: arXiv:2503.11733.

[6] S. Conklin, T. Dorgan, D. Barreto: *Is AI the new course creator*, Discov Educ 3 (2024), p. 285, doi: 10.1007/s44217-024-00386-2.

[7] C. Deng et al.: *Deconstructing The Ethics of Large Language Models from Long-standing Issues to New-emerging Dilemmas: A Survey*, 2024, doi: 10.48550/arXiv.2406.05392, eprint: arXiv:2406.05392.

[8] B. Denny, J. Prather, B. A. Becker, J. Finnie-Ansley, A. Hellas, J. Leinonen, A. Luxton-Reilly, B. N. Reeves, E. A. Santos, S. Sarsa: *Computing education in the era of generative AI*, Communications of the ACM 67.2 (2024), pp. 56–67, doi: 10.1145/3624720.

[9] J. Finnie-Ansley, P. Denny, B. Becker, A. Luxton-Reilly, J. Prather: *The robots are coming: Exploring the implications of OpenAI Codex on introductory programming*, in: Proceedings of the 24th Australasian Computing Education Conference, 2022, pp. 10–19, doi: 10.1145/3511861.3511863.

[10] S. García-Méndez, F. de Arriba-Pérez, M. d. C. Somoza-López: *A Review on the Use of Large Language Models as Virtual Tutors*, Sci & Educ 34 (2025), pp. 877–892, doi: 10.1007/s11191-024-00530-2.

[11] N. Gardella, R. Pettit, S. L. Riggs: *Performance, workload, emotion, and self-efficacy of novice programmers using AI code generation*, in: Proceedings of the 2024 on Innovation and Technology in Computer Science Education, 2024, pp. 290–296, doi: 10.1145/3649217.3653615.

[12] S. Groothuijsen, A. Beemt, J. C. Remmers, L. W. Meeuwen: *AI chatbots in programming education: Students' use in a scientific computing course and consequences for learning*, Computers and Education: Artificial Intelligence 7 (2024), p. 100290, doi: 10.1016/j.caeai.2024.100290.

[13] Q. Huang, C. Lv, L. Lu, T. Shuang: *Evaluating the Quality of AI-Generated Digital Educational Resources for University Teaching and Learning*, Systems 13.3 (2025), p. 174, doi: 10.3390/systems13030174.

[14] D. M. Johnson, W. Doss, C. M. Estepp: *Using ChatGPT with novice Arduino programmers: Effects on performance, interest, self-efficacy, and programming ability*, Journal of Research in Technical Careers 8.1 (2024), doi: 10.9741/2578-2118.1152.

[15] M. Kazemitabaar, J. Chow, C. K. T. Ma, B. J. Ericson, D. Weintrop, T. Grossman: *Studying the effect of AI code generators on supporting novice learners in introductory programming*, in: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 2023, doi: 10.1145/3544548.3580919.

[16] L. Labadze, M. Grigolia, L. Machaidze: *Role of AI chatbots in education: systematic literature review*, Int J Educ Technol High Educ 20 (2023), p. 56, doi: 10.1186/s41239-023-00426-1.

[17] M. Lepp, J. Kaimre: *Does generative AI help in learning programming: Students' perceptions, reported use and relation to performance*, Computers in Human Behavior Reports 18 (2025), p. 100642, doi: 10.1016/j.chbr.2025.100642.

[18] Y. Liu, T. Le-Cong, R. Widyasari, C. Tantithamthavorn, L. Li, X. B. D. Le, D. Lo: *Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues*, ACM Transactions on Software Engineering and Methodology 33.5 (2024), pp. 1–26, DOI: 10.1145/3643674.

[19] B. Ma, L. Chen, S. Konomi: *Enhancing Programming Education with ChatGPT: A Case Study on Student Perceptions and Interactions in a Python Course*, 2024, DOI: 10.48550/arXiv.2403.15472, eprint: arXiv:2403.15472.

[20] A. Parasuraman, C. Colby: *An Updated and Streamlined Technology Readiness Index: TRI 2.0*, Journal of Service Research 18.1 (2015), DOI: 10.1177/1094670514539730.

[21] A. Park, T. Kim: *Code suggestions and explanations in programming learning: Use of ChatGPT and performance*, The International Journal of Management Education 23.2 (2025), p. 101119, DOI: 10.1016/j.ijme.2024.101119.

[22] J. Prather, B. N. Reeves, J. Leinonen, S. MacNeil, A. S. Randrianasolo, B. A. Becker, B. Kimmel, J. Wright, B. Briggs: *The widening gap: The benefits and harms of generative AI for novice programmers*, in: Proceedings of the 2024 ACM Conference on International Computing Education Research - Volume 1, 2024, pp. 469–486, DOI: 10.1145/3632620.3671116.

[23] M. M. Rahman, Y. Watanobe: *ChatGPT for education and research: Opportunities, threats, and strategies*, Applied Sciences 13.9 (2023), p. 5783, DOI: 10.3390/app13095783.

[24] M. M. Rahman, Y. Watanobe, R. U. Kiran, R. Kabir: *A stacked bidirectional lstm model for classifying source codes built in mpls*, in: Proceedings of the Machine Learning and Principles and Practice of Knowledge Discovery in Databases: International Workshops of ECML PKDD 2021, 2021, pp. 75–89, DOI: 10.1007/978-3-030-93733-1_5.

[25] M. M. Rahman, Y. Watanobe, K. Nakamura: *A neural network based intelligent support model for program code completion*, Sci. Program. 2020 (2020), p. 7426461, DOI: 10.1155/2020/7426461.

[26] M. M. Rahman, Y. Watanobe, K. Nakamura: *Source code assessment and classification based on estimated error probability using attentive LSTM language model and its application in programming education*, Appl. Sci. 10 (2020), p. 2973, DOI: 10.3390/app10082973.

[27] P. Rajabi, P. Taghipour, D. Cukierman, T. Doleck: *Unleashing ChatGPT's impact in higher education: Student and faculty perspectives*, Computers in Human Behavior: Artificial Humans 2.2 (2024), p. 100090, DOI: 10.1016/j.chbah.2024.100090.

[28] D. Ravšelj, D. Keržič, N. Tomaževič, L. Umek, N. Brezovar, N. A. Iahad, et al.: *Higher education students' perceptions of ChatGPT: A global study of early reactions*, PLoS ONE 20.2 (2025), e0315011, DOI: 10.1371/journal.pone.0315011.

[29] M. Richards, K. Waugh, M. Slaymaker, M. Petre, J. Woodthorpe, D. Gooch: *Bob or bot: Exploring ChatGPT's answers to university computer science assessment*, ACM Transactions on Computing Education 24.5 (2024), pp. 1–32, DOI: 10.1145/3633287.

[30] A. Saranti, B. Taraghi, M. Ebner, A. Holzinger: *Property-based testing for parameter learning of probabilistic graphical models*, in: Lncs: 12279. CD-MAKE 2020, ed. by A. Holzinger, P. Kieseberg, A. M. Tjoa, E. Weippl, 2020, pp. 499–515, DOI: 10.1007/978-3-030-57321-8_28.

[31] A. Šarčević, I. Tomičić, A. Merlin, M. Horvat: *Enhancing Programming Education with Open-Source Generative AI Chatbots*, in: 2024 47th MIPRO ICT and Electronics Convention (MIPRO), Opatija, Croatia, 2024, pp. 2051–2056, DOI: 10.1109/MIPRO60963.2024.10569736.

[32] D. Sobania, M. Briesch, F. Rothlauf: *Choose your programming copilot: A comparison of the program synthesis performance of github copilot and genetic programming*, in: Proceedings of the Genetic and Evolutionary Computation Conference, New York, NY, USA, 2022, pp. 1019–1027, DOI: 10.1145/3512290.3528700.

[33] S. Steinert, K. E. Avila, S. Ruzika, J. Kuhn, S. Küchemann: *Harnessing large language models to develop research-based learning assistants for formative feedback*, Smart Learn. Environ. 11 (2024), p. 62, doi: 10.1186/s40561-024-00354-1.

[34] D. Sun, A. Boudouaia, C. Zhu, Y. Li: *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 (2024), p. 14, doi: 10.1186/s41239-024-00446-5.

[35] Y. Xue, H. Chen, G. R. Bai, R. Tairas, Y. Huang: *Does ChatGPT Help With Introductory Programming? An Experiment of Students Using ChatGPT in CS1*, in: ICSE-SEET '24: Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training, 2024, pp. 331–341, doi: 10.1145/3639474.3640076.

[36] R. Yilmaz, F. G. K. Yilmaz: *The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation*, Computers & Education: Artificial Intelligence 4 (2023), doi: 10.1016/j.caeai.2023.100147.

[37] Q. Zhang, C. Fang, Y. Ma, W. Sun, Z. Chen: *A Survey of Learning-based Automated Program Repair*, 2023, doi: 10.1145/3631974, eprint: arXiv:2301.03270.