International Journal of Innovative Research and Scientific Studies, 8(5) 2025, pages: 157-167

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

GenAI potential and challenges in programming knowledge control

Zhanat Nurbekova[1*], Ainur Sagimbayeva[2], Asset Zhaxylykov[3], Esen Bidaibekov[4]

[1,2,4]*Abai Kazakh National Pedagogical University, Almaty, Kazakhstan.*
[3]*Toraighyrov University, Pavlodar, Kazakhstan.*

Corresponding author: Zhanat Nurbekova (*Email: zhasin2006@gmail.com*)

Abstract

The purpose of this article is to critically analyze the potential and challenges associated with the use of GenAI (generative artificial intelligence) in the automated verification and formation of students' programming knowledge assessments. In terms of methodology, this study employs a mixed approach: analyzing the functionality of popular generative models and platforms for competitive programming (Codeforces, LeetCode, HackerRank), as well as comparing their applicability to various task formats and types of knowledge (syntactic, semantic, logical). The possibilities of such systems in providing adaptive feedback, automating routine checks, and supporting individualized training are considered. It was revealed that, despite advantages such as scalability, interactivity, and the ability to generate contextually relevant content, there are significant limitations, including risks of incorrect assessment, vulnerability to model deception, lack of pedagogical interpretation of results, and potential dependence of students on AI assistance. In conclusion, the need to develop regulatory, pedagogical, and technical frameworks that ensure the ethical and responsible application of GenAI in programming knowledge assessment systems is emphasized.

Keywords: Automated assessment, Formative assessment, GenAI, GitHub Copilot, Individualization of learning, Knowledge control, Programming, ChatGPT.

DOI: 10.53894/ijirss.v8i5.8577
Funding: This work is supported by the Committee of Science of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant Number: AP23490592).
History: Received: 28 May 2025 / **Revised:** 3 July 2025 / **Accepted:** 7 July 2025 / **Published:** 16 July 2025

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

In the digital age, programming training at the university plays an important role in the formation of students' professional skills, ensuring their competitiveness in the labor market, and contributes to the development of innovative technologies. Programming training at a university should take into account modern trends in the IT industry, as well as combine theoretical knowledge with practical skills. Programming is a complex area, the development of which requires systematic efforts, a special technique, and the development of various skills. Therefore, it is necessary to develop more comprehensive, effective, and universal educational and methodological tools for teaching and controlling students' programming knowledge [1].

With the development of information technology and the advent of online platforms, the process of teaching and testing students' programming knowledge has changed significantly. Modern Codeforces, LeetCode, HackerRank, and other platforms for competitive programming not only help students develop their skills but also provide opportunities to apply them by solving problems and participating in competitions. Automated assessment on online platforms often emphasizes functional correctness, neglecting other important aspects such as code quality and style. This limited feedback may prevent students from understanding and mastering programming concepts, thereby reducing the overall educational value of the learning experience [2]. Students have difficulty interpreting error messages, leading to repeated errors in submissions for automated validation on a platform with programming tasks. In addition, the lack of distinction between student and tester code errors complicates understanding, affecting learning outcomes and overall performance when evaluating programming [3].

Despite these challenges, the potential for automated assessment systems to improve the learning process remains significant, especially as technology continues to evolve and address these challenges.

The integration of GenAI into online learning and programming knowledge assessment presents significant potential. GenAI can provide individual feedback and resources, enhancing the process of personalized learning and monitoring students' programming knowledge [4]. Students use GenAI tools to create code and explanations, optimizing the learning process and enhancing programming knowledge. Additionally, GenAI enables educators to effectively support large groups of students by offering ideas that were previously unattainable [5].

The teaching practices of the authors of the article on programming at universities demonstrate that GenAI has an increasing impact on assessment processes and learning outcomes, which are interconnected.

The aim of this study is to analyze various online platforms with programming tasks and the impact of GenAI on methods for monitoring and evaluating students' knowledge, and to develop recommendations for improving these methods based on existing approaches.

Based on the goal, the following research questions are posed:

1. What methods of automated knowledge assessment are used on online programming platforms, and how can GenAI improve them?

2. What challenges do online platforms face when implementing GenAI to control knowledge, and how can risks be minimized?

3. How can online platforms use GenAI to personalize learning and increase the effectiveness of knowledge testing?

2. Materials and Methods

Traditional methods for assessing students' programming knowledge include various approaches such as tests, projects, verbal surveys, and competitive assignments. These methods are essential in both traditional and online educational settings, often adapted to enhance learning outcomes. Test tasks evaluate students' theoretical knowledge and programming skills, considering their level of training. They offer a structured way to assess understanding of programming concepts and principles [6]. Project assignments enable students to apply their knowledge to practical scenarios by developing problem-solving and project management skills. They are effective in assessing programming skills [7, 8]. Oral surveys can improve their critical thinking and communication skills required by programming professionals [9]. Competitive assignments encourage innovation and collaboration, motivating students to apply their skills in real-world settings, which is vital for their professional development [8].

Despite the steady spread of traditional methods, much attention is paid to the integration of technology into knowledge control processes, which contributes to the development of more personalized and effective mechanisms for controlling programming knowledge. This demonstrates the transformation of the educational environment, in which automated systems become an integral addition to classical approaches. For example, Learning Management Systems (LMS), online courses, online tutoring platforms, and adaptive learning platforms allow you to personalize and adapt content and activities to the individual needs of students. They can offer individualized learning trajectories, adaptive content, and individualized delivery of learning materials [10].

Note that the format of online platforms with programming tasks and algorithms such as Codeforces, LeetCode, HackerRank, and others plays an important role in teaching programming. They provide a structured environment for learning and practicing programming skills. These platforms also offer many features that consider various aspects of learning, from problem solving to gamification and content integration.

Codeforces' online platform infrastructure supports the entire learning process, from task creation to competition, making it a valuable tool for educators [11]. The integration of Codeforces into educational programs enhances student engagement and achievement, particularly in programming courses [12]. The lack of personalized feedback in some of its functions may complicate the learning process for students who require more guidance [13].

The online platform with programming tasks LeetCode has attracted attention for its role in assessing the capabilities of artificial intelligence models in solving programming problems. Studies have shown that models such as GPT-4 can solve problems with high accuracy on medium-level LeetCode, although their performance may fluctuate with subsequent attempts [14]. This platform is often used to evaluate LLM performance, and studies show that these models can generate efficient and correct code [15]. The integration of AI tools with LeetCode has also improved the learning experience for students (LeetCode AI Chrome extension). The research further examined the use of HackerRank as a tool for evaluating artificial intelligence models. For example, the Codex model demonstrated uniqueness in solving problems in Python on HackerRank, achieving 96% success in settings with zero attempts and 100% in settings with multiple attempts [16].

Table 1 presents a comparative analysis of online platforms with programming tasks and algorithms. It should be noted that the platforms under consideration for competitive programming, due to algorithmic tasks of varying complexity, contribute to the development of practical programming skills and stimulate independent learning through the possibility of choosing an individual set of tasks and participating in competitions. However, they do not have the function of adaptive learning.

Table 1.
Comparative analysis of online platforms Code forces, Leet Code, Hacker Rank.

Platform	Advantages	Participation of expert teachers
Codeforces	Monitors the progress of training and compares the results of students. Regularly holds contests ("rounds"), providing students with the opportunity to improve their programming skills in real time. It is aimed at a wide audience and offers level-appropriate tasks. The platform database contains more than 5,000 tasks covering various topics in the field of programming. An active learning community promotes sharing experiences, discussing solutions, and collaborative learning.	Personalization of learning plays an important role in learning programming. For beginners, the choice of suitable tasks can be difficult, so the teacher selects tasks corresponding to the student's level of training. Complex tasks require additional explanations that can be given in training sessions. Formative knowledge control helps to timely identify the discrepancy between the complexity of tasks and the level of training of the student, which contributes to the adjustment of training and reduces the likelihood of difficulties in the decision process. The teacher coordinates the educational process, analyzing the content of tasks and selecting additional educational materials. Interaction with the teacher ensures the adaptation of tasks taking into account the individual characteristics and level of training of each student.
LeetCode	The platform covers a wide range of topics, including both basic data structures and complex algorithmic concepts such as dynamic programming, graphs, search, and sorting algorithms. Users are given a choice between free access and a paid subscription, which unlocks additional features, including exclusive tasks and detailed analysis of solutions. Additionally, the platform offers videos and articles to help users better understand how to solve problems. Regular contests and a rating system provide users with opportunities to compare their results, develop professional skills, and participate in a competitive environment.	The teacher plays an important role in complementing learning, directing students to perform specific tasks to better master certain topics. Analysis of successful solutions and discussion of strategies contribute to the formation of logic for solving problems and the development of analytical thinking among students. In addition, the instructor helps compensate for the limited focus of the training by expanding the study of topics beyond interview preparation, including the analysis of architectural solutions and a deeper understanding of complex concepts.
HackerRank	A programming learning platform offering a wide range of categories including algorithms, data structures, databases, artificial intelligence, and other areas. Focused on interview preparation, it provides tasks that mimic real-world interviews at top companies like Google and Facebook. Flexible task formats and support for multiple programming languages allow users to choose the most convenient language for solving problems. Regular competitions and an active community contribute to the exchange of experience and increased motivation. The rating and statistics system helps to track progress and achieved results.	Teaching support on the HackerRank platform promotes effective learning by combining practical tasks with theoretical material to better understand concepts. The teacher can optimize the complexity of tasks by selecting more complex tasks for experienced students or analyzing them in class. The use of ratings and statistics allows for adjusting the individual learning trajectory, taking into account the progress and characteristics of each student.

In the context of the development of online education, platforms are effective tools for controlling programming knowledge, but they are not able to replace the role of a teacher. Interaction with the instructor is necessary to personalize training, adapt tasks, analyze errors, and explain complex concepts. The active participation of the teacher provides a deeper understanding of the material and the adjustment of the learning trajectory, especially when working with difficult tasks.

The functionality of online platforms for teaching programming has significantly improved educational techniques and student engagement. Educational platforms provide interactive environments that facilitate real-time feedback, critical thinking, and personalized learning experiences as needed to master programming concepts. Virtual laboratory platforms (VLPs) significantly improve programming skills compared to traditional methods, contributing to a better understanding of logic, syntax, and code structure [17].

Multifunctional platforms support a wide range of programming disciplines, providing students with the opportunity to study using task-oriented approaches and various coding tools [18].

One of the functionals of the platform is the assessment of the work of students, which provides feedback [19]. Knowledge control mechanisms monitor student progress and generate reports, facilitating personalized learning [10].

Some platforms include tools for analyzing the spatiotemporal complexity of algorithms, helping students understand the effectiveness of their code [20].

Most of the platforms reviewed include functions such as formal assessments and certification upon course completion, confirming the skills acquired by students [21]. There are also functions with elements of gamification, performance analytics, and access to the global learning community, which increase engagement and motivation [22].

Thus, the functionality of online platforms for teaching programming can include: assessment of programming knowledge - automatic verification of the code for correctness, optimality, and compliance with the requirements of the task; evaluation types - checking the level of material development using tests, tasks, and quizzes; coding assistance - built-in hints, code completion, access to documentation; support for many programming languages - Python, Java, C++, JavaScript, etc.; flexibility and multitasking - the ability to work on multiple projects, integration with external tools; coding workspace - built-in code editor with syntax highlighting and debugging; training levels - tasks of different complexity and materials depending on the training level; complexity analysis - estimation of execution time and memory usage by the algorithm; code review and optimization - automatically identifying errors and recommending performance improvements; exercises and practice sections - interactive tasks and projects to consolidate theory; assessment and certification - obtaining certificates after completing courses or tests; additional opportunities and functions - gamification, social features, competitions, and team projects; support and training - reference documentation, video manuals, and technical support.

Based on the functionality identified, an analysis of online programming learning platforms was performed (Figure 1).

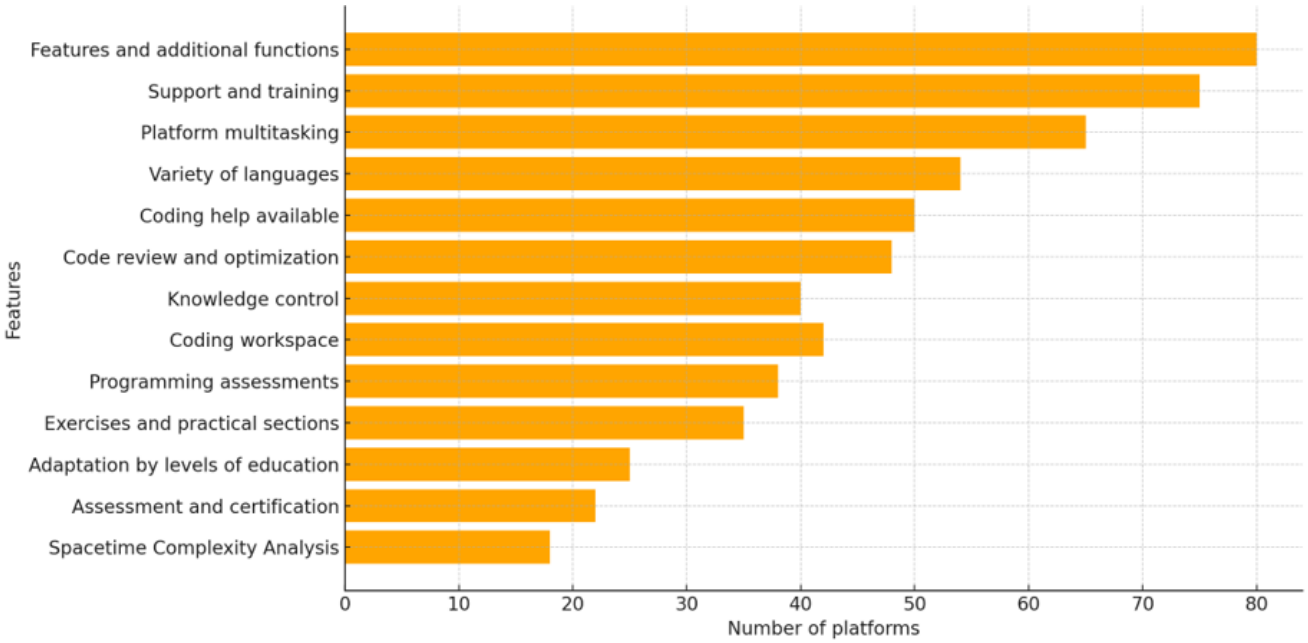

Figure 1.
Functionality of online platforms for programming training.

Analysis of the presented functionality of online platforms Figure 1 shows that knowledge control is widely represented but inferior to leading capabilities. Encoding assistance is common, but not all systems have active prompts. Code validation and optimization are common but not key for all platforms. At the same time, the assessment of programming knowledge is present in a significant number of platforms, which confirms its importance in the educational process.

Analyzing the methodology of knowledge control in online programming platforms (LeetCode, Codeforces, HackerRank), the following key aspects were identified: types of knowledge control, assessment criteria, task complexity levels, additional features, and time for checking problem solutions Figure 2.

Each platform offers different levels of complexity (from easy to difficult), which allows you to flexibly adapt the learning process to the individual needs of students. Thus, a comparative analysis of online platforms demonstrates the need for a differentiated approach to training and knowledge control in programming, depending on the goals and level of student training. Understanding the features of each platform helps you choose the most suitable one for a specific purpose: interview, competition, or training.

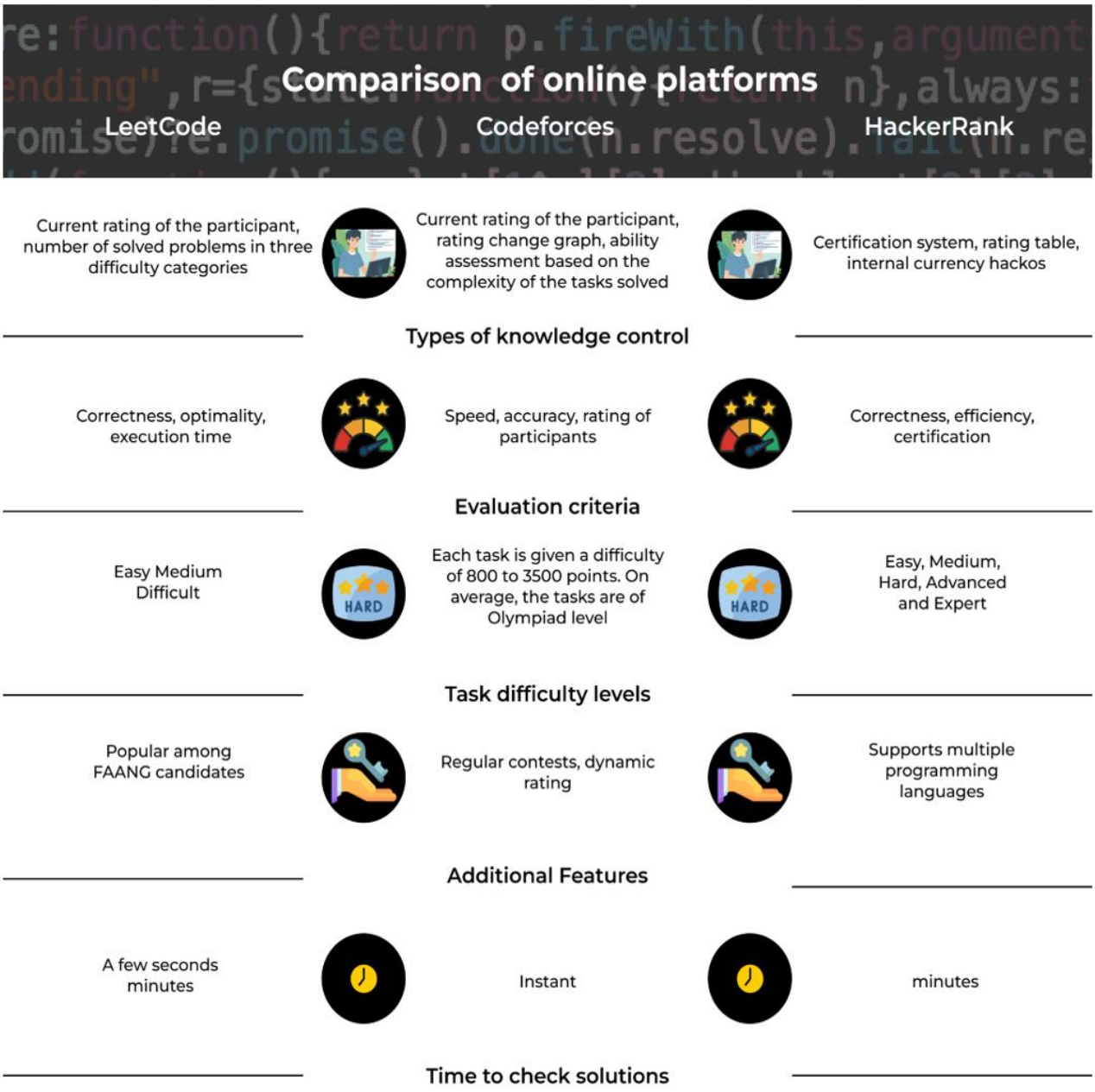

Figure 2.
Methodology for knowledge control in the online platforms LeetCode, Codeforces, HackerRank

Knowledge control on online platforms requires not only an analysis of the platforms' functionality but also an assessment of the content presented. This approach enables the identification of key aspects that teachers should focus on, even when an automated knowledge assessment system is employed. Table 2 presents the key aspects of knowledge control in programming, as well as the role of online platforms and the teacher.

Table 2.
Key Aspects of Programming Knowledge Control and the Role of Online Platforms and Instructor.

Aspects of Programming Knowledge Control	Role of the online platform	Instructor role
Parsing task conditions	Automated tests check the correctness of the answer, but do not help the student understand the conditions.	Helps to interpret the task, explains the key points.
Data structure selection	The platform does not analyze which data structure was selected, but only the result.	Explains why one option is better than another, teaches you to analyze complexity.
Algorithm optimality	Evaluates only on a run-time basis, but does not analyze the code itself.	Helps evaluate different approaches, identify flaws in logic.
Debugging and Error Analysis	The automated system can indicate that the answer is incorrect, but does not explain why	Explains errors, helps find weaknesses in the code.
Adapting Task Complexity	Platforms give tasks of a random level of difficulty, but do not always correctly assess a student's training.	The teacher can select tasks corresponding to the level of the student, adjust the difficulty.
Link to real projects	Tasks are often divorced from real cases.	Shows how knowledge is applied in real-world development

Table 3 presents an analysis of the knowledge control methodology on online platforms.

Table 3.
Analysis of the knowledge control methodology on online platforms.

Types of control	Description	Scoring metrics
Contest-Based Assessment	Knowledge assessment through competitions, rating system (Codeforces, TopCoder)	Rating, solution success, execution speed
Test-Based Assessment	Evaluation of solutions on test data, automated testing (LeetCode, AtCoder)	Percentage of correct decisions, number of attempts
Project-Based Assessment	Knowledge assessment based on project execution, code review (Coursera, Kaggle)	Code quality, meeting project requirements
Formative Assessment	Interim assessment of knowledge with feedback (Codeforces Gym, Stepik)	Learning progress, mistakes in intermediate tasks

Thus, the content analysis of programming courses allowed us to identify the strengths and weaknesses of existing online platforms and suggest possible areas for their improvement.

To effectively control knowledge on online platforms, you need to focus on developing students' programming skills, so you should integrate critical thinking, employment skills, and systematic content analysis into the curriculum.

Interactive and query-based learning methods are necessary to develop analytical skills and practical problem-solving abilities [23].

A rigorous approach to online platform content analysis is necessary to ensure the reliability and validity of educational assessments. The use of structured methodologies can improve the quality of knowledge control and align educational outcomes with industry standards.

While focusing on skills development is vital, it is also important to recognize that traditional testing methods can still play a role in assessing basic knowledge. Balancing both approaches can yield the best educational outcomes.

Analysis of the content of online platforms revealed how fully and effectively the system teaches students, what aspects require adjustment, and where and how the teacher should be involved. It is important to understand what to focus on when controlling knowledge so that the platform does not just test students but also develops their skills. Along with this, we conclude the need to involve a teacher to adjust the programming training process. Because automated knowledge control does not replace the analysis of solutions, discussion of alternative and individual approaches to solving the problem. The teacher plays a key role in personalizing the task by difficulty levels, helping to analyze, adjust the training trajectory, and adapt the training to the students.

Integrating AI into online platforms enhances the educational experience by personalizing learning, automating assessments, and increasing engagement. The use of AI technologies such as neural networks and natural language processing enables real-time feedback and the development of individualized learning trajectories, which are essential for effective programming education.

Online platforms can adapt to individual learning styles by providing personalized content and recommendations based on user performance [24]. Intelligent programming assistants can compensate for faculty shortages by offering personalized support and real-time interaction, which promotes student motivation [25]. Neural networks automate code analysis, provide recommendations for improvement, and offer real-time feedback, significantly enhancing the learning experience for students and reducing teachers' review time [26]. AI is currently partially integrated into online programming platforms, but its use is limited to automatic code checking, task selection, and ratings. In the future, there are likely to be more

advanced AI assistants who will be able to explain errors, adapt training, and give personalized recommendations, which will make the process of learning programming even more efficient Table 4.

Table 4.
AI Inspection Types.

Types of control	Using AI	Problems	Optimization
Contest-Based Assessment)	Selection of rating tasks, automatic error analysis	Does not take into account the level of training, and reading is possible	Taking into account the individual level, adaptive selection of tasks
Test-Based Assessment	Automatic solution validation, improvement recommendations	Does not evaluate code style, restricted validation criteria	Add Code Style Score, Enhanced Test Suites
Project-Based Assessment	Feedback generation, code quality analysis	Time-consuming to check, difficult to automate	Automated code review, using AI assessment
Formative Assessment	Personalized recommendations, error-based hints	Dependence on feedback quality, complexity of personalization	Flexible recommendation system, integration with AI assistants

Since programming proceeds from the task in the process of modeling network data, it is necessary to assess the level of application of graph structures. In order to determine the skills of managing the order of processing tasks, it is necessary to assess the ability to use stacks and queues [2]. Knowledge control on online platforms should consider the student's level of understanding of various data structures (arrays, queues, stacks, graphs, etc.). It is important to understand how thoroughly each structure is assessed by the platform, as well as which aspects require additional teacher intervention Figure 3.

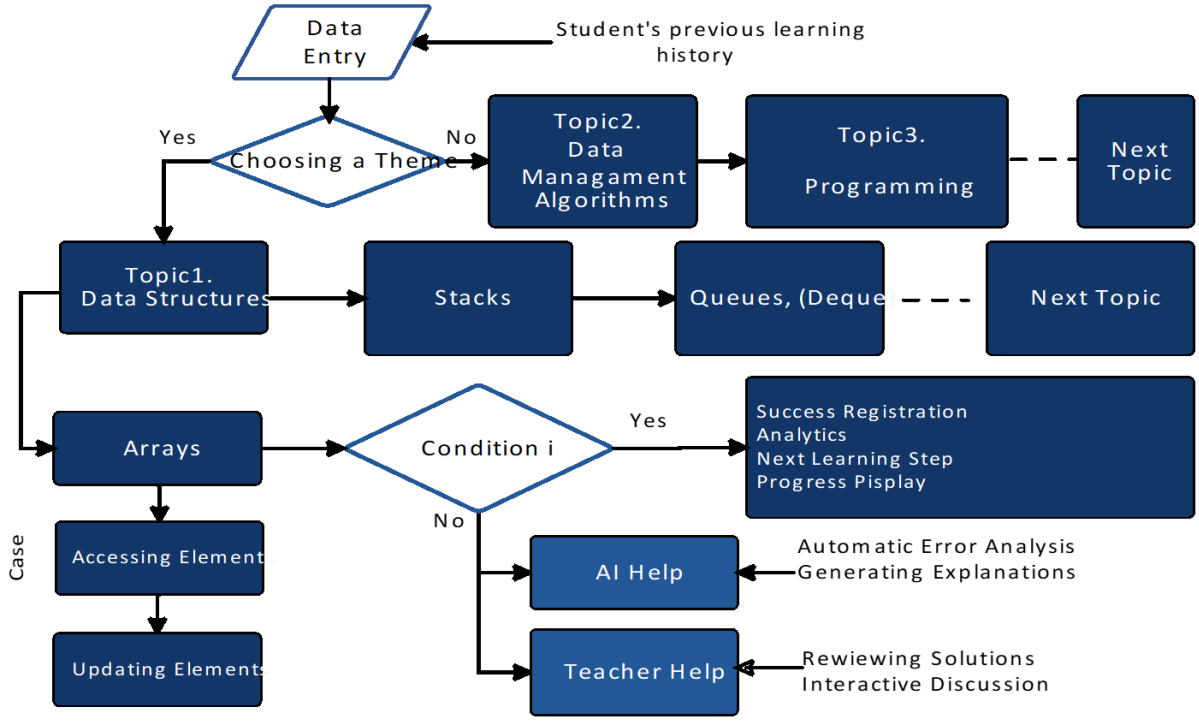

Figure 3.
Algorithm for monitoring students' programming knowledge. On the example of a topic on data structures.

This flowchart illustrates the process of monitoring students' programming knowledge using online platforms (for example, Codeforces, LeetCode, HackerRank) with the integration of AI assistance and teacher support to ensure effective testing of knowledge, optimize the learning process, and support students in case of difficulties.

Data entry (initial stage): This block is responsible for collecting input information: student data, previous training history, level of training, and information that comes from the platform through a stop test or input by a teacher.

Topic selection: At this stage, the topic is determined, within which the student will solve problems. The topic is selected based on the student's previous successes, individual level of knowledge, and educational plan.

Topic-specific subtopic refinement: after selecting a primary theme (for example, Data Structure), the Array subcategories are refined to verify array skills, including accessing and updating elements. Stack: Checks the ability to manage the data stack. Queue (Deque): Checks the correct use of queues and double-ended queues, etc.

Checking the fulfillment of conditions (Condition I): In this block, an automated check of the solution is carried out for compliance with the specified criteria. Main criteria include: correctness of the solution, all tests are passed; code efficiency, satisfaction of time and memory restrictions; compliance with the requirements of the task, choosing the correct data structure and algorithm. If all conditions are met, progress is recorded in the system, the rating or points increase, and a more complex task or a new topic is proposed. If the conditions are not met, the student is given the opportunity to use: AI automatic error analysis, optimal solution, generation of clarifications and recommendations. With the help of a teacher: individual consultation, analysis of complex decisions, detailed analysis of errors, and comments.

Thus, a flexible approach to knowledge control with the possibility of both automatic and teaching support. This structure allows you to quickly respond to errors and effectively adjust the educational process. Using AI and teaching assistance in the complex, you can achieve a higher quality of learning and prepare students for real tasks in programming.

Solutions to unique problems on the online platforms Codeforces, LeetCode, and HackerRank are analyzed. Each of these platforms employs various methods of knowledge assessment, including automated testing, rating systems, and competitive design formats. The levels of test tasks range from basic to advanced, allowing for training adaptation to different user skill levels. Evaluation criteria are based on decision correctness, algorithm efficiency, and code optimality. Verification time depends on automatic tests or manual moderation, providing operational feedback. Overall, the analysis indicates that knowledge assessment on these platforms contributes to an objective evaluation of skills and enhances the quality of programming training.

3. Results

Implementing GenAI on online platforms to control knowledge presents a number of challenges, most notably the spread of misleading information, student engagement, and maintaining critical thinking. Addressing these challenges requires a multifaceted approach to minimize associated risks.

GenAI can create highly convincing but false content, leading to the spread of misinformation. Platforms must refine their existing moderation policies to effectively address this problem [27]. GenAI has the potential to facilitate productive discussions; however, it can also lead to superficial interactions if not managed properly. Platforms need frameworks to ensure that AI-generated content facilitates meaningful dialogue [28]. Over-reliance on GenAI can diminish students' critical thinking skills, as they may depend excessively on results generated by GenAI. This necessitates the development of GenAI systems that promote critical evaluation of information [29].

On the one hand, platforms must strengthen their content moderation rules to cover content generated by AI, treating it in the same way as traditional malicious content [27]. The introduction of frameworks that guide the use of GenAI in discussions may help preserve the quality of discourse and ensure that GenAI serves as a tool for engagement rather than a substitute for human interaction [28]. The development of GenAI interfaces that encourage critical analysis and reflection may reduce the risks of cognitive decline associated with over-reliance on GenAI tools [29].

While GenAI integration offers significant opportunities to improve knowledge management and user experience, it is critical to remain vigilant about potential pitfalls. Balancing AI capabilities with human control is essential to take advantage of it while minimizing risks [30].

GenAI enables significant improvements to online platforms through personalized learning and more effective knowledge control. Using GenAI, it is possible to tailor the educational environment to the individual needs of learners, thereby promoting engagement and academic performance.

GenAI can personalize learning based on individual learner outcomes and preferences, ensuring that each learner has an individual educational experience [31]. Tools such as QuizGPT can create tests of different difficulty levels, adapting to the learner's knowledge and skills, which improves comprehension and memorization [32].

Personalized GenAI-based assessments can maximize student motivation and engagement, enabling students to demonstrate their knowledge more effectively [33].

GenAI can provide real-time feedback on assessments, helping students identify areas for improvement and adjust their learning strategies accordingly [31].

GenAI facilitates the implementation of Universal Design for Learning (UDL) principles by addressing diverse learning needs, thereby promoting inclusivity in online education. By personalizing learning materials and assessments, GenAI can increase empathy and understanding, making learning more accessible to all students [34].

While there are numerous benefits to integrating GenAI into education, it also raises concerns about data privacy, algorithmic bias, and the potential dehumanization of learning experiences. Balancing these challenges with the benefits of personalized education remains critical for future developments in this area [35].

4. Discussions

The application of GenAI in controlling students' programming knowledge opens new possibilities but also presents challenges for educators. The primary issue is the discrepancy between the capabilities of these tools and the educational objectives of the courses, which complicates their integration into the educational process [36].

In addition, the use of GenAI complicates the objective assessment of knowledge, since students can rely on generated solutions rather than understanding tasks on their own [37]. This also increases the risk of plagiarism; students may pass off code created by GenAI as their own without citing the source.

Another challenge is the possible bias of GenAI algorithms, which can lead to unequal learning conditions [37]. Faculty also note that overuse of GenAI may reduce the level of critical thinking and self-reliance in students [38]

However, with proper application, GenAI can be a valuable learning aid. It is able to provide personalized feedback and increase student engagement. By finding a balance between traditional methods and GenAI technologies, you can minimize risks and effectively utilize its advantages.

The use of GenAI in programming assignments presents a number of challenges for students. According to surveys, 48.5% of students regularly use GenAI for homework, which leads to excessive dependence on technology [39]. This practice can impede the development of fundamental skills in solving algorithmic problems, causing difficulties at more advanced stages of training, which in pedagogical literature is referred to as the "junior course wall" [39].

Despite the high potential of models such as GPT-4, they remain prone to errors in various scenarios, which can mislead students and contribute to uncritical decision-making [40]. In addition, the quality of the generated code varies, which increases the likelihood of errors that students may not recognize or understand [41].

The availability of automated solutions reduces students' motivation to study programming concepts deeply, which negatively affects the development of the discipline [42]. Thus, educators need to develop strategies to integrate GenAI into the educational process while maintaining the development of critical thinking and practical coding skills [42].

Let's consider the problems that students face when solving programming problems using GenAI. While GenAI can help create code and provide explanations, it can also lead to superficial learning and cognitive dissonance among aspiring programmers. Many novice programmers do not have metacognitive strategies, which can be exacerbated by the use of GenAI. Students may find it difficult to comprehend how their dependence on AI affects their problem-solving abilities [43]. Struggling students often experience the illusion of competence, believing that they understand the material better than they actually do, which may interfere with their learning process [43].

Over-reliance on GenAI may reduce students' autonomy and critical thinking skills, as they may become dependent on AI for solutions rather than developing their own problem-solving strategies [44].

The need for human control is critical, as students cannot always recognize the accuracy of AI-generated content, leading to potential misinformation [45].

Effective collaboration with GenAI requires students to develop communication strategies, which can be difficult for those unfamiliar with AI tools. Individual differences in learning styles may further complicate this collaboration [46].

In contrast, some trainees may find that GenAI improves their learning experience by providing immediate feedback and support, suggesting that the impact of GenAI is not uniquely negative. Balancing AI assistance with traditional teaching methods can produce the best outcomes for students.

4. Conclusion

In conclusion, we note that the competent use of GenAI in teaching programming can become an effective tool for personalized learning, providing immediate feedback and adaptive teaching methods, which in turn contribute to improving the quality of the educational process.

However, it should be noted that using GenAI for programming assignments presents a number of difficulties for students, especially in terms of metacognitive awareness, critical thinking, and reliance on technology.

The analysis shows that without transparent criteria, the use of GenAI can lead to incorrect results. In addition, built-in algorithmic biases in AI models are possible. If students use AI as a problem solver, then algorithmic thinking and programming skills in students could be adversely affected.

GenAI should be used as a tool to accompany and provide feedback, serving as a substitute for reflection and critical analysis. It should be integrated into learning didactics to individualize instruction, analyze errors, and adjust the learning trajectory. AI-integrated systems must guarantee the protection of students' personal data and their work, providing mechanisms to distinguish independent work from generated work. These systems must be sustainable and compatible with existing educational platforms.

Thus, in order to eliminate the risk of substitution of real automation training, the risk of loss of confidence in the control and assessment system, and the risk of ethical violations, it is necessary to create regulatory, pedagogical, and technical frameworks that ensure the ethical and reasonable application of GenAI in programming knowledge control systems.

References

[1] C. S. Cheah, "Factors contributing to the difficulties in teaching and learning of computer programming: A literature review," *Contemporary Educational Technology*, vol. 12, no. 2, p. ep272, 2020. https://doi.org/10.30935/cedtech/8247

[2] A. Hameer and B. Pientka, "Teaching the art of functional programming using automated grading (experience report)," *Proceedings of the ACM on Programming Languages*, vol. 3, no. ICFP, pp. 1-15, 2019. https://doi.org/10.1145/3341719

[3] S. Caton, S. Russell, and B. A. Becker, "What fails once, fails again: Common repeated errors in introductory programming automated assessments," in *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education-Volume 1*, 2022, pp. 955-961.

[4] L. Yan, S. Greiff, Z. Teuber, and D. Gašević, "Promises and challenges of generative artificial intelligence for human learning," *Nature Human Behaviour*, vol. 8, no. 10, pp. 1839-1850, 2024. https://doi.org/10.48550/arXiv.2408.12143

[5] J. Prather *et al.*, "Beyond the hype: A comprehensive review of current trends in generative ai research, teaching practices, and tools," *2024 Working Group Reports on Innovation and Technology in Computer Science Education*, pp. 300-338, 2025. https://doi.org/10.48550/arxiv.2412.14732

[6] Meenu, "Assessment methods: Traditional exams vs. continuous assessment," *Universal Research Reports*, vol. 11, no. 5, pp. 26–31, 2024. https://doi.org/10.36676/urr.v11.i5.1447

[7] A. P. Deliwe and Z. Zvapano, "A model for effective implementation of contemporary assessment methods in higher education institutions," *Incremental Cost-Effectiveness Ratio*, vol. 1, no. 1, 2024. https://doi.org/10.34190/icer.1.1.2874

[8] D. M. Djamalovna, "Methods and tools for assessing student competencies," *Current Research Journal of Philological Sciences*, vol. 5, no. 10, pp. 19-24, 2024. https://doi.org/10.37547/philological-crjps-05-10-04

[9] A. Şişianu and A. Pușcașu, "Methods of assessing students' knowledge in english language lessons," *Journal of Social Sciences*, vol. 3, pp. 234-242, 2024.

[10] J. Anghelo, M. C. Bedoya-Flores, E. F. Mosquera-Quiñonez, Á. E. Mesías-Simisterra, and J. V. Bautista-Sánchez, "Educational platforms: Digital tools for the teaching-learning process in education," *Ibero-American Journal of Education & Society Research*, vol. 3, no. 1, pp. 259-263, 2023. https://doi.org/10.56183/iberoeds.v3i1.626

[11] M. Mirzayanov *et al.*, "Codeforces as an educational platform for learning programming in digitalization," *Olympiads in Informatics*, vol. 14, 2020. https://doi.org/10.15388/ioi.2020.10

[12] Y. Zhang, Z. Li, B. Du, Y. Wu, and H. Jiang, "Data analysis of online judge system-based teaching model," in *International Conference on Computer Science and Education*, 2022: Springer, pp. 531-543.

[13] M. Amirabbas, M. Vahidi-Asl, A. Khalilian, A. Baraani-Dastjerdi, and B. Zamani, "Code4Bench: A multidimensional benchmark of Codeforces data for different program analysis techniques," *Journal of Computer Languages*, vol. 53, pp. 38-52, 2019. https://doi.org/10.5281/zenodo.2582967

[14] S. Vishnu, Sahil, and N. Garg, "Unveiling the role of gpt-4 in solving leetcode programming problems," *Computer Applications in Engineering Education*, vol. 33, no. 1, p. e22815, 2025. https://doi.org/10.1002/cae.22815

[15] T. Coignion, C. Quinton, and R. Rouvoy, "A performance study of llm-generated code on leetcode," in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, 2024, pp. 79-89.

[16] A. Karmakar, J. A. Prenner, M. D'Ambros, and R. Robbes, "Codex hacks hackerrank: Memorization issues and a framework for code synthesis evaluation," *arXiv preprint arXiv:2212.02684*, 2022. https://doi.org/10.48550/arXiv.2212.02684

[17] I. Maulana, E. Supriyadi, and B. S. Wijanarka, "Coding learning innovation: Interactive programming experience with virtual lab platform," *Multidisciplinary Science Journal*, vol. 7, no. 6, pp. 2025262-2025262, 2025. https://doi.org/10.31893/multiscience.2025262

[18] A. A. Nikandrov, "Multifunctional and flexible online platforms for creating educational materials," *Informatika i Obrazovanie*, vol. 37, no. 6, pp. 22-29, 2023. https://doi.org/10.32517/0234-0453-2022-37-6-22-29

[19] N. Torres and P. Olivares, "Impact of an interactive learning platform on programming student outcomes: a case study of teloprogramo," in *2024 43rd International Conference of the Chilean Computer Science Society (SCCC)*, 2024: IEEE, pp. 1-7.

[20] F. E. Qodirov, G. U. Nazarova, and M. A. Qurbonova, "The use of educational software and tools for teaching programming," *International Journal of Innovative Science and Research Technology*, pp. 1981-1984, 2024. https://doi.org/10.38124/ijisrt/ijisrt24oct1769

[21] N. V. Krasilnikova and S. V. Saraikina, "Educational platforms as an opportunity for personal and professional self-realization of students," *Research Result. Pedagogy and Psychology of Education*, vol. 9, no. 3, 2023. https://doi.org/10.18413/2313-8971-2023-9-3-0-2

[22] A. Nachankar, "Smart ICT fusion hub," *International Journal For Science Technology And Engineering*, vol. 12, no. 12, pp. 792-795, 2024. https://doi.org/10.22214/ijraset.2024.65902

[23] M. Jamil, A. Jan, and N. Muhammad, "Pre-service teachers' perceptions of pedagogical skills learned during teacher education program," *Pakistan JL Analysis & Wisdom*, vol. 3, p. 163, 2024.

[24] A. Manoharan and G. Nagar, "Maximizing learning trajectories: an investigation into AI-driven natural language processing integration in online educational platforms," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 3, pp. 01-10, 2021. https://doi.org/10.56726/irjmets18093

[25] H. Wu *et al.*, "Research on the construction of intelligent programming platform based on AI-Generated content," in *Proceedings of the 15th International Conference on Education Technology and Computers*, 2023, pp. 9-15.

[26] O. Koval, "Application of artificial intelligence in programming education within a blended learning environment," *Open educational E-environment of Modern University*, vol. 17, pp. 65-78, 2024. https://doi.org/10.28925/2414-0325.2024.175

[27] S. A. Fisher, J. W. Howard, and B. Kira, "Moderating synthetic content: The challenge of generative AI," *Philosophy & Technology*, vol. 37, no. 4, p. 133, 2024. https://doi.org/10.1007/s13347-024-00818-9

[28] L. L. Tsai, A. Pentland, A. Braley, N. Chen, J. R. Enríquez, and A. Reuel, "Generative AI for pro-democracy platforms," in *Proceedings of the ACM Symposium on Computing for Social Impact*, 2024. https://doi.org/10.21428/e4baedd9.5aaf489a.

[29] A. Sarkar, N. Toronto, I. Drosos, and C. Poelitz, "When copilot becomes autopilot: generative ai's critical risk to knowledge work and a critical solution," *arXiv preprint arXiv:2412.15030*, 2024. https://doi.org/10.48550/arxiv.2412.15030

[30] M. Alavi, D. Leidner, and R. Mousavi, "Knowledge management perspective of generative artificial intelligence (GenAI)," *Journal of the Association for Information Systems*, pp. 1-12, 2024.

[31] Y. Singh, D. P. Singh, N. Chander, Y. P. Singh, and P. R. Singh, "Generative AI for enhancing education and skill development," *International Journal of Innovative Science and Research Technology*, pp. 883-888, 2024. https://doi.org/10.38124/ijisrt/ijisrt24nov431

[32] C.-K. Chang, "Enhancing academic performance with generative ai-based quiz platform," in *2024 IEEE International Conference on Advanced Learning Technologies (ICALT)*, 2024: IEEE, pp. 193-195.

[33] B. Arslan *et al.*, "Opportunities and challenges of using generative AI to personalize educational assessment," *Frontiers in Artificial Intelligence*, vol. 7, p. 1460651, 2024. https://doi.org/10.3389/frai.2024.1460651

[34] A. Morgan, "Leveraging generative artificial intelligence to expedite UDL implementation in online courses," in Unlocking Learning Potential With Universal Design in Online Learning Environments. Hershey, PA: IGI Global, 2024, pp. 171-195.

[35] J. Hutson and D. Plate, *Leveraging AI to personalize and humanize online learning in unlocking learning potential with universal design in online learning environments*. Hershey, PA: IGI Global, 2024. https://doi.org/10.4018/979-8-3693-0762-5.ch011

[36] C. Zastudil, M. Rogalska, C. Kapp, J. Vaughn, and S. MacNeil, "Generative AI in computing education: perspectives of students and instructors," in *2023 IEEE Frontiers in Education Conference (FIE)*, 18-21 Oct. 2023 2023, pp. 1-9. https://doi.org/10.1109/fie58773.2023.10343467.

[37] W. K. Monib, A. Qazi, R. A. Apong, M. T. Azizan, L. De Silva, and H. Yassin, "Generative AI and future education: A review, theoretical validation, and authors' perspective on challenges and solutions," *PeerJ Computer Science*, vol. 10, p. e2105, 2024. https://doi.org/10.7717/peerj-cs.2105

[38] Y. Talgatov, G. K. Kassymova, and M. Nurtanto, "AI in the classroom: A boon or a threat to pedagogical practices?," in *Proceedings of the 2024 International Conference on Education and Pedagogical Science (pp. 128-134)*, 2024. https://doi.org/10.31643/2024.19.

[39] E. Dickey, A. Bejarano, and C. Garg, "Innovating Computer Programming Pedagogy: The AI-Lab Framework for Generative AI Adoption," *arXiv preprint arXiv:2308.12258*, 2023. https://doi.org/10.48550/arxiv.2308.12258

[40] T. Phung *et al.*, "Generative AI for programming education: benchmarking ChatGPT, GPT-4, and human tutors," in *Proceedings of the 2023 ACM Conference on International Computing Education Research-Volume 2*, 2023, pp. 41-42, doi: https://doi.org/10.48550/arXiv.2306.17156

[41] A. M. Dakhel, A. Nikanjam, F. Khomh, M. C. Desmarais, and H. Washizaki, "Generative AI for software development: A family of studies on code generation," in Generative AI for Effective Software Development, A. Nguyen-Duc, P. Abrahamsson, and F. Khomh Eds. Cham: Springer Nature Switzerland, 2024, pp. 151-172. https://doi.org/10.1007/978-3-031-55642-5_7

[42] B. A. Becker, P. Denny, J. Finnie-Ansley, A. Luxton-Reilly, J. Prather, and E. A. Santos, "Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation," in *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, 2023, pp. 500-506. https://doi.org/10.48550/arxiv.2212.01020

[43] J. Prather *et al.*, "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.

[44] J. Zhang, "Generative ai in higher education: Challenges and opportunities for course learning," *Advances in Social Sciences Research Journal*, vol. 12, no. 01, pp. 11-18, 2025. https://doi.org/10.14738/assrj.1201.18121

[45] N. Mellqvist and P. Mozelius, "Exploring student perspectives on generative AI in requirements engineering education," *Proceedings of the International Conference on AI Research*, vol. 4, no. 1, pp. 473-481, 2024. https://doi.org/10.34190/icair.4.1.3136

[46] W. Yan, T. Nakajima, and R. Sawada, "Benefits and challenges of collaboration between students and conversational generative artificial intelligence in programming learning: An empirical case study," *Education Sciences*, vol. 14, no. 4, p. 433, 2024. https://doi.org/10.3390/educsci14040433