Environment. Technology. Resources. Rezekne, Latvia
Proceedings of the 16th International Scientific and Practical Conference. 2025. Volume III, 253-259

Artificial Intelligence as a Support Tool in Teaching Programming to Future Bachelor's Students of Vocational Education

Bohdan Rozputnia
Vinnytsia Mykhailo Kotsiubynskyi State Pedagogical University
Vinnytsia, Ukraine
b.rozputnia@vspu.edu.ua

Lyudmila Shevchenko
Vinnytsia Mykhailo Kotsiubynskyi State Pedagogical University
Vinnytsia, Ukraine
shevchenko@vspu.edu.ua

Volodymyr Umanets
Vinnytsia Mykhailo Kotsiubynskyi State Pedagogical University
Vinnytsia, Ukraine
umanets@vspu.edu.ua

Serhii Yashchuk
Vinnytsia Mykhailo Kotsiubynskyi State Pedagogical University
Vinnytsia, Ukraine
serhii.yashchuk@vspu.edu.ua

Yuliia Sabadosh
Vinnitsya National Technical University
Vinnytsia, Ukraine
avataric87@gmail.con

***Abstract*- The article presents a thorough examination of the potential applications of artificial intelligence (AI) in supporting the instruction of programming to future bachelor's degree students in vocational education. It explores the pivotal domains of AI integration into the educational process, encompassing the utilization of adaptive learning systems, intelligent tutoring systems, automated code evaluation systems, and generative models that enhance both the theoretical and practical training of students. It demonstrates the ways in which AI enhances the personalization of educational content, facilitates rapid feedback loops, and optimizes the verification process of software solutions. It has been determined that the integration of AI facilitates the creation of adaptive learning environments. In such environments, automated algorithms analyze test results, the history of students' interaction with educational materials, and the personal pace of information assimilation. Consequently, this facilitates the development of customized educational pathways that are tailored to the distinct characteristics of each student. The implementation of intelligent tutoring systems, such as ChatGPT, GitHub Copilot, or Google AI Studio based on Gemini, facilitates the elucidation of complex programming concepts, including the principles of recursion, sorting algorithms, and other fundamental principles. This, in turn, contributes to the cultivation of critical thinking and self-study skills. In the article, the authors analyze the challenges associated with the introduction of AI in the educational process. The primary challenges identified pertain to issues of academic integrity, particularly when future bachelors of vocational education employ AI capabilities to automatically generate solutions without a comprehensive grasp of the subject matter. Additionally, the article addresses technical limitations concerning the substantial computing resources required and the integration of contemporary algorithms into existing educational platforms. The article further underscores the necessity for specialized professional development programs to equip educators with the skills to effectively utilize AI in vocational education. Additionally, it emphasizes the establishment of ethical frameworks to guide the implementation of AI technologies in this context, ensuring that the principles of academic integrity are preserved and the integrity of the educational process is maintained. The authors of the article propose a number of recommendations and approaches to optimize the process of AI integration, create integrated learning environments, and improve existing assessment methods with regard to automated code verification. The findings of the study can be utilized to enhance pedagogical approaches in programming, to improve the quality of training in the field of information technology, and to promote the development of competitive graduates.**

***Keywords*- Artificial Intelligence, vocational education, programming training, adaptive learning systems.**

I. INTRODUCTION

In the contemporary context, marked by the proliferation of digital technologies and the rapid transformation of the labor market, there is an imperative to enhance the pedagogical approaches employed in the training of future bachelor's degree holders in vocational education specializing in programming.Conventional

Online ISSN 2256-070X
https://doi.org/10.17770/etr2025vol3.8537

teaching methodologies encounter a myriad of challenges, including a paucity of personalization, an absence of prompt feedback, and the inability to effectively validate students' software solutions.In this milieu, the integration of AI holds considerable potential to transform the learning process by rendering it adaptive, efficient, and technologically agile.

Specifically, AI stands as a pioneering instrument capable of substantially enhancing the caliber of programming education.The field of AI encompasses the development of intelligent machines and systems, endowed with the capacity to execute tasks formerly relegated to human intelligence. Such tasks include learning, data analysis, pattern recognition, decision-making, and natural language processing. The field of AI is founded on machine learning, deep learning, and other methodologies that empower systems to adapt to novel data and enhance their performance without the necessity of explicit programming [1]. The capabilities of adaptive learning, automated code review, the generation of personalized tasks, and instant feedback are instrumental in facilitating the creation of an effective learning environment that caters to the distinct needs and training levels of each student. The integration of AI in programming education has been shown to facilitate knowledge assimilation, cultivate critical thinking skills, and enhance self-study abilities, as well as the capacity to swiftly adapt to emerging technologies [2].

Moreover, the integration of AI into the educational process is congruent with global trends in education, including the transition to digital and interactive teaching methods. This is of particular significance for the training of future professionals, who must be prepared to function in a context of constant innovation and change. Consequently, the exploration of the potential of leveraging AI as a pedagogical tool to support the instruction of programming for future bachelor's degree programs in vocational education is of paramount importance. This initiative is directed towards addressing the pressing challenges encountered in contemporary education and enhancing the employability of graduates in the labor market.

II. MATERIALS AND METHODS

The integration of AI into the educational process has been widely discussed in recent research, with a particular focus on its application in programming instruction. Studies by Luo et al. and Zawacki-Richter et al. emphasize the potential of AI to enhance personalized learning, optimize assessment methods, and reduce the workload of educators through automated grading and knowledge adaptation. AI-driven systems facilitate predictive analytics, enabling the identification of students requiring additional support, and contribute to the development of adaptive learning environments that adjust task complexity based on individual proficiency levels.

This study employs a systematic literature review and comparative analysis of AI-powered educational tools to evaluate their role in programming education for future bachelor's degree students in vocational education. The literature review includes peer-reviewed articles indexed in Scopus, IEEE Xplore, and Web of Science, with a focus on research published between 2019 and 2024. Particular attention is given to studies by Shevchenko et al. and Lytvynova et al., which explore the effectiveness of AI in automating programming task evaluation and providing tailored recommendations for code improvement. Additionally, the study examines practical AI implementations such as intelligent tutoring systems, automated assessment platforms, and AI-assisted integrated development environments.

The research methodology also involves a qualitative analysis of AI applications in programming education, drawing on previous empirical findings. The study evaluates AI's impact on student engagement, learning efficiency, and problem-solving skills while considering potential challenges related to academic integrity, instructor preparedness, and technological constraints. The findings contribute to a broader understanding of AI's role in vocational education and provide recommendations for its effective implementation in programming instruction.

III. RESULTS AND DISCUSSION

A review of the extant scientific literature reveals the considerable promise of AI in the domain of programming education. Notwithstanding, concerns regarding methodological support, the development of effective adaptive learning algorithms, and the ethical considerations of AI in the educational process persist. AI has been demonstrated to play a pivotal role in enhancing the learning of theoretical programming fundamentals through adaptive learning, intelligent tutoring systems, and automated support for students [1]-[5].

The integration of AI within educational settings has been shown to enhance personalization of learning experiences, boost student engagement, and alleviate cognitive load by automating the delivery of explanations and feedback. A particularly salient domain within the realm of AI in education is that of Intelligent Tutoring Systems (ITS), which are designed to provide customized assistance to students.Empirical evidence has demonstrated the efficacy of ITS in emulating teacher behaviors, assessing student progress, and tailoring explanations to align with each student's level of understanding [7].

AI assistants such as ChatGPT, DeepSeek, and Gemini have the capacity to respond to students' inquiries in real time, elucidating theoretical programming concepts in a comprehensible manner. These assistants employ Natural Language Processing (NLP) and machine learning algorithms to generate comprehensive explanations of intricate subjects.For instance, a student can inquire about the principles of recursion, and the system will not only define the concept but also furnish code examples, identifying the aspects that may present a significant challenge for a specific user.

To demonstrate the potential of AI-driven tutoring systems in programming education, the authors have

developed a specialized prompt designed to enhance AI-assisted instruction. This prompt enables AI assistants to provide structured, beginner-friendly explanations tailored to vocational education, integrating theoretical knowledge with practical applications.

For instance, when teaching conditional statements (if, else, elif) in Python, the AI is instructed to define the concept in simple terms, present a workplace-related example—such as automating a temperature check in a manufacturing setting—and offer a relevant code snippet. The AI then encourages the student to apply the concept by composing a conditional statement for a vocational task, such as verifying safety protocols. If difficulties arise, the AI breaks the problem into smaller steps and provides corrective feedback with step-by-step explanations, reinforcing the connection between programming skills and their future profession.

The following is an excerpt from the AI prompt developed in this study:

"*You are an AI assistant designed to support students in learning programming for their future careers. Your task is to explain programming concepts in a clear, beginner-friendly manner, provide real-world examples relevant to vocational tasks (e.g., automation, data analysis), and guide students through hands-on coding exercises. Use an encouraging tone, correct mistakes with step-by-step explanations, and link concepts to practical job-related applications.*

Focus on conditional statements (if, else, elif) in Python. Start by defining the concept in simple terms. Provide a workplace example (e.g., automating a temperature check in a manufacturing environment). Include a code snippet. Then, ask the student to write a conditional statement for a vocational scenario (e.g., verifying safety protocols). If they struggle, break the problem into smaller steps. Highlight how mastering this skill applies to their future profession."

This structured interaction demonstrates how AI-driven assistants can bridge the gap between theoretical programming knowledge and industry-relevant applications, fostering a more intuitive and career-oriented learning experience for vocational students (Listing 1).

```
# CONDITIONAL STATEMENTS IN PYTHON
# Scenario: Temperature Monitoring in
Manufacturing
# Why this matters: Automating safety
checks reduces human error.
# Simulate a machine's current temperature
(in Celsius)
current_temp = 145
# Check if temperature exceeds safe upper
limit (150°C)
if current_temp > 150:
    print("⚠️ Warning: Temperature too
high! Shutting down.")  # Trigger
emergency protocol
# Check if temperature is below safe lower
limit (100°C)
elif current_temp < 100:
    print("⚠️ Warning: Temperature too
low! Adjust settings.")  # Alert for
calibration

# If temperature is within safe range
else:
    print("✅ Temperature normal.
Operation continues.")  # Confirm stable
conditions
# STUDENT TASK: SAFETY PROTOCOL
VERIFICATION
# Vocational context:
Manufacturing/Industrial Settings
# Goal: Ensure two safety protocols are
met before machine startup.
# Step 1: Define safety status variables
(modify values to test)
goggles_on = True          # Safety
goggles must be worn
emergency_stop_working = True  # Emergency
button must be functional
# Step 2: Write a conditional to check
BOTH protocols
# Hint: Use 'and' to require both
conditions to be True
if goggles_on and emergency_stop_working:
    # Display success message if both are
True
    print("✅ Safety protocols met.
Machine starting.")
else:
    # Display failure message if any
condition fails
    print("❌ Safety check failed.
Investigate and retry.")

# Career connection: Such checks prevent
accidents in factories.
```

Listing 1. Prompt result illustrates how AI-powered tutoring can support students in mastering conditional statements by providing clear explanations.

A study by Zawacki-Richter et al. demonstrates that the incorporation of intelligent tutoring systems into the educational process enhances learning efficiency. This enhancement is attributed to the ability of AI to provide customized explanations and immediate feedback, representing a substantial advancement over conventional teaching methods.

The implementation of adaptive learning, facilitated by AI, enables the adjustment of the complexity of theoretical materials to align with the individual student's level of knowledge. This enhancement in learning efficacy is attributable to the analysis of test results, the history of interaction with educational materials, and the rate of information assimilation.To illustrate, in the event that an intelligent tutoring system detects a student's inability to comprehend the principles of loops in the Python programming language, the system can offer

supplementary explanations, interactive exercises, or alternative approaches to learning the topic.The utilization of AI in this context is predicated on the application of machine learning and cluster analysis methods to identify patterns in students' learning achievements.

This approach enables the creation of customized learning pathways, a critical factor in the development of programming competencies.The study by Luo et al. corroborates the efficacy of adaptive learning systems leveraging AI in adapting learning resources to the distinct needs of students, thereby enhancing the quality of learning [2].

The integration of AI within programming education has been identified as a significant advancement, particularly in the context of error correction. This capability is particularly beneficial during the initial stages of learning, when students frequently encounter challenges in identifying syntactic and logical errors in code. According to a study by Lytvynova et al. the incorporation of AI into Integrated Development Environments (IDEs) facilitates automatic code analysis and the provision of immediate error explanations [5].

For instance, AI-enabled development environments (e.g., JetBrains AI Assistant, GitHub Copilot) offer explanations for specific code fragments, recommendations for optimizing algorithms, and even step-by-step instructions for fixing errors.It is noteworthy that contemporary educational platforms such as CodeSignal, Mimir Classroom, and CodeGrade employ AI algorithms to analyze students' code in real time.

This facilitates not only the evaluation of task accuracy but also the generation of comprehensive reports that assess the strengths and weaknesses of the code, thereby enhancing learning efficiency.Intelligent tutoring systems, adaptive platforms, and automated hints serve to mitigate barriers to learning complex concepts by offering more effective feedback and a more flexible management of the learning process. Notwithstanding the numerous benefits, the quality of automatically generated explanations, the level of academic integrity, and the need to train teachers to effectively utilize AI in teaching remain significant concerns [6].

The integration of AI into the educational process has emerged as a significant paradigm shift, with the potential to enhance the learning of theoretical programming fundamentals. This integration encompasses the implementation of adaptive learning methodologies, intelligent tutoring systems, and automated support for students, thereby personalizing the educational experience and enhancing student engagement. The incorporation of AI-driven systems automates the delivery of explanations and feedback, thereby reducing the cognitive load on students and facilitating a more efficient learning environment.The assessment of learning outcomes and the provision of effective feedback constitute pivotal aspects of cultivating students' professional competencies.

In the context of traditional programming education systems, the process of code review and error analysis is often labor-intensive and time-consuming for teachers. The integration of AI to automate these processes can enhance efficiency, reduce grading delays, and personalize the educational experience.

While AI-driven automation addresses inefficiencies in instructor-led evaluation, its potential extends beyond administrative streamlining. Modern AI tools, such as conversational assistants (e.g., ChatGPT, Gemini) and code-centric platforms (e.g., GitHub Copilot, JetBrains AI Assistant), further enrich programming education by providing dual-axis support: they alleviate instructor workload and empower students through real-time, context-aware guidance. These tools diverge in functionality—conversational AI excels at conceptual scaffolding and adaptive problem-solving, while code-centric systems optimize practical implementation and syntax precision.

The following analysis categorizes their distinct educational applications (Table 1), demonstrating how their complementary roles bridge theoretical understanding and vocational skill development. For instance, AI not only automates error detection but also transforms mistakes into teachable moments, fostering iterative learning cycles that mirror real-world software debugging processes.

TABLE I. COMPARATIVE ANALYSIS OF AI TOOLS FOR PROGRAMMING EDUCATION

Service	Main Features	Advantages
ChatGPT/ Gemini	Natural language interaction, concept explanation, debugging guidance, personalized learning support.	Accessibility, adaptability to diverse skill levels, promotes self-directed learning.
GitHub Copilot	Code autocompletion, real-time suggestions, IDE integration, multi-language support.	Enhances productivity, reduces syntax errors, ideal for project-based learning.
JetBrains AI Assistant	Context-aware code generation, refactoring tools, error detection within IDEs.	Seamless integration with professional tools, supports advanced vocational workflows.

A study [8] found that the integration of AI in the evaluation of practical tasks has a positive impact on students' academic performance. This is due to the immediate feedback provided by AI systems, which enables students to swiftly correct errors. Additionally, AI systems can adapt the complexity of tasks to suit the unique needs of each student, enhancing the efficiency and personalization of the learning.

Automated assessment systems facilitate not only the evaluation of task correctness but also the assessment of programming style, the efficiency of algorithms, and the alignment of code with contemporary standards [8]. For

instance, CodeGrade integrates with Learning Management Systems (LMS) to automatically analyze code, verify its alignment with course requirements, and generate reports for students and teachers.

A study by Dunder et al. demonstrated that the integration of AI into automated assessment systems leads to a substantial reduction in teachers' workload, enabling them to allocate more time to methodological and advisory activities.Moreover, the automation of the assessment process fosters objectivity and standardization, which are crucial factors in ensuring the quality of education [7].

One of the primary benefits of incorporating AI within the context of programming education is the capacity to provide immediate feedback. AI-based tools are capable of not only identifying errors in code but also elucidating their underlying causes and proposing potential solutions. For instance, GitHub Copilot analyzes code and offers recommendations for optimization, while JetBrains AI Assistant integrates with integrated development environments (IDEs) to assist with programming writing and debugging.

A study by Frankford et al. found that students who utilize AI for automated assessment demonstrate accelerated progress in learning programming languages. This is due to the ability to promptly obtain explanations for errors and make corrections [9]. However, it is important to note that excessive reliance on AI can diminish the level of independent critical thinking. Therefore, the effective use of these technologies necessitates a balanced approach that incorporates automated support with personal code analysis.

Notwithstanding its myriad advantages, the utilization of artificial intelligence in the domain of code review automation is encumbered by certain limitations. The primary limitations pertain to the following:

1. The potential for erroneous evaluation of complex algorithms: Contemporary AI systems demonstrate proficiency in recognizing syntactic and logical errors. However, they may erroneously interpret complex algorithmic structures and innovative approaches to problem-solving.

2. Concerns regarding academic integrity: Students may employ AI to automatically generate responses without comprehending the theoretical underpinnings, thereby jeopardizing profound learning.

3. The necessity to adapt curricula is also imperative, as the integration of AI into the educational process necessitates alterations in teaching methodologies and educational materials, which in turn requires the appropriate training of teachers.In our estimation, the introduction of AI in automated code review and feedback represents a promising domain of educational technology development with the potential to considerably enhance the efficiency of programming education. Concomitantly, it is imperative to take into account potential challenges and devise strategies to overcome them in order to ensure a balanced utilization of these technologies.

The integration of AI into the professional education of future bachelor's degree programs in vocational education is a promising area, but it is accompanied by a number of challenges and limitations. Despite significant advancements in the field of AI development, technical limitations may affect its effectiveness in teaching programming. For instance, studies have demonstrated that contemporary models such as ChatGPT can effectively solve simple tasks but encounter challenges when confronted with more complex programming problems. In the study titled "Kattis vs. ChatGPT: Assessment and Evaluation of Programming Tasks in the Age of Artificial Intelligence," [7] ChatGPT was able to solve only 19 out of 127 programming tasks provided by the Kattis automatic evaluation system, indicating its limited capabilities in complex scenarios.

However, the rapid evolution of AI technologies suggests that such findings may quickly become outdated as newer, more capable models emerge. Research by Jošt et al. highlights this shift, demonstrating that modern large language models LLMs exhibit enhanced performance in programming education, particularly when utilized as support tools [10]. Their study, involving undergraduate students learning React applications, found that reliance on LLMs for complex tasks, such as code generation and debugging, negatively correlated with final grades (Spearman's rho = -0.305, p = 0.045; rho = -0.360, p = 0.021), whereas their use for supplementary explanations showed no significant adverse effect (rho = -0.201, p = 0.135) [1]. These results suggest a growing potential for LLMs in vocational education, provided their application is judiciously balanced.

Nevertheless, the findings of earlier studies, such as the 2019 ChatGPT evaluation, should not be overemphasized, as they reflect AI capabilities at a specific point in time. Jošt et al. emphasize that the advancing capabilities of LLMs necessitate a reevaluation of their educational role [10]. As AI progresses, with models demonstrating increased proficiency in addressing complex programming challenges, educators must adapt pedagogical strategies to leverage these tools effectively. This approach ensures that AI enhances, rather than overshadows, the development of essential programming skills, aligning with the evolving demands of vocational education.

The integration of AI into the educational process necessitates a reevaluation of conventional pedagogical approaches. Teachers must adapt their teaching methods to effectively utilize AI as a support tool. underscores the imperative to develop novel teaching strategies that take into account the capabilities and limitations of AI, as well as the necessity to train teachers to utilize these technologies. The incorporation of AI in education gives rise to concerns regarding ethics and privacy. The collection and analysis of student data may raise concerns about privacy and security of information. Furthermore,

there is a risk of students' overdependence on AI, which may negatively affect the development of their critical thinking and independent problem-solving skills. The studys [7]-[9] points to the need to balance the use of AI and traditional teaching methods to ensure the comprehensive development of students.

The implementation of AI necessitates substantial resources, including advanced computing systems and access to voluminous data sets. Not all educational institutions possess the requisite infrastructure and financial resources to undertake such endeavors, which can result in disparities in access to contemporary technologies among students across different institutions.The rapid advancements in AI technologies demand incessant updating of teaching materials and adaptation to novel tools and methodologies.Teachers must be prepared to engage in continuous learning and enhancement of their skills to effectively integrate AI into the educational process.

The advent of AI as a pedagogical tool for the instruction of programming in the context of vocational education has the potential to significantly enhance the efficacy of the educational process. However, the effective integration of AI technologies necessitates further research, the adaptation of curricula, and the implementation of innovative teaching methodologies. The development of AI in programming education presents several key prospects, among which are the following.

The development of adaptive educational platforms leveraging AI will facilitate the precise alignment of educational material to the distinct needs of each student.Contemporary systems already employ machine learning methods to assess students' knowledge and progress; however, their capabilities can be augmented by incorporating more sophisticated neural networks and prediction algorithms [8].

It is anticipated that future advancements in AI-powered assistants within the domain of programming education will extend beyond mere code explanations to encompass the identification of optimal learning strategies for each student. These strategies will be determined through comprehensive analysis of students' historical experience and cognitive styles.Platforms such as Microsoft's IntelliCode and JetBrains' AI assistants are progressively aligning with this direction by incorporating progressively sophisticated intelligent features, thereby providing enhanced support to novice programmers.

AI has already been incorporated into the development of automated code review systems, yet there is ample room for enhancement.Presently, the evaluation of code quality and compliance with best programming practices is primarily conducted by existing platforms, which focus on syntactic and logical correctness. However, future advancements may encompass the analysis of code quality and adherence to optimal programming methodologies [9].

The subsequent phase may entail the creation of AI systems that not only verify the correctness of code but also educate students by offering recommendations for enhancing the efficiency, readability, and optimization of code.The integration of generative models such as GPT-4, Gemini, and Claude into programming education represents a promising avenue for advancement. These models have the potential to function not only as assistants for writing code, but also as interactive tutors that elucidate programming concepts, assist with code debugging, and model real-world software development scenarios.For instance, the integration of generative AI into IDEs (Integrated Development Environments) has the capacity to enable students to receive instant code explanations and predictions of possible errors before the program is launched.This approach has the potential to markedly increase the efficiency of learning and render the process of learning programming more intuitive.

In light of the extant literature on the subject, the following recommendations are put forth for the advancement of AI integration in the realm of programming education:

1. The establishment of integrated learning environments that seamlessly integrate adaptive courses with automated code verification mechanisms.

2. The development of assessment methods that leverage AI capabilities, including the verification of the uniqueness of program logic.

3. The implementation of professional development programs for educators that will empower them to utilize AI in an effective manner within the educational process.

4. The development of national policies on the use of AI in professional education to regulate academic integrity and ethics of AI use.In conclusion, the integration of AI into programming education holds great promise; however, a comprehensive approach and close cooperation between technology developers, teachers, and educational researchers are essential for its successful implementation.

IV. CONCLUSIONS

This study underscores that the incorporation of AI into vocational programming education holds considerable transformative potential, enabling a shift from traditional, one-size-fits-all instruction to a more personalized and adaptive learning paradigm.The analysis indicates that AI can significantly enhance learning efficiency and student engagement by facilitating real-time, individualized feedback and dynamically tailoring educational content.

However, the findings also highlight the need for a balanced implementation approach that addresses inherent challenges, such as ensuring academic integrity, overcoming technical limitations, and equipping educators with the necessary skills to harness these technologies effectively. Rather than merely automating existing processes, the successful integration of AI should be viewed as part of a broader pedagogical reform.This reform involves revising curricula to incorporate AI-driven assessment tools and adaptive learning environments, alongside establishing robust ethical frameworks and professional development programs.

The study underscores the necessity for additional empirical research to validate these strategies and to refine the interplay between technology, instructional methods, and policy-making.In summary, while AI offers promising avenues for enriching programming education, its long-term success depends on an integrated, multi-faceted strategy that aligns technological innovation with sound pedagogical practices and ethical oversight.

REFERENCES

[1] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed. Pearson, 2020.

[2] Z. Luo, J. Jiang, and H. Zhou, "Exploration on the application of artificial intelligence technology in vocational education teaching," J. High. Vocat. Educ., vol. 1, no. 2, pp. 177–186, 2024.

[3] O. Zawacki-Richter, V. I. Marín, M. Bond, and F. Gouverneur, "Systematic review of research on artificial intelligence applications in higher education – where are the educators?," Int. J. Educ. Technol. High. Educ., vol. 16, no. 39, 2019, doi: 10.1186/s41239-019-0171-0. [Online]. Available: https://doi.org/10.1186/s41239-019-0171-0

[4] L. S. Shevchenko, V. O. Umanets, and B. M. Rozputnia, "Applying generative AI to automate teacher tasks in vocational education," Open Educ. E-Environ. Mod. Univ., vol. 17, pp. 160–170, 2024, doi: 10.28925/2414-0325.2024.1711.

[5] S. Lytvynova, N. Rashevska, and S. Proskura, "The use of artificial intelligence in teaching students programming languages," Inst. Digit. Educ., Nat. Acad. Educ. Sci. Ukraine, 2024.

[6] V. Kyslitsyn, L. Shevchenko, V. Umanets, L. Sikoraka, and Y. Angelov, "Applying the Python programming language and Arduino robotics kits in the process of training future teachers of computer science," ETR, vol. 2, pp. 162–167, Jun. 2024. [Online]. Available: https://doi.org/10.17770/etr2024vol2.8026.

[7] N. Dunder, S. Lundborg, O. Viberg, and J. Wong, "Kattis vs. ChatGPT: Assessment and evaluation of programming tasks in the age of artificial intelligence," arXiv preprint, arXiv:2312.01109, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2312.01109.

[8] T. Wang, D. Vargas-Díaz, C. Brown, and Y. Chen, "Exploring the role of AI assistants in computer science education: Methods, implications, and instructor perspectives," arXiv preprint, arXiv:2306.03289, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2306.03289.

[9] E. Frankford, C. Sauerwein, P. Bassner, S. Krusche, and R. Breu, "AI-Tutoring in Software Engineering Education," arXiv preprint, arXiv:2404.02548, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2404.02548.

[10] G. Jošt, V. Taneski, and S. Karakatič, "The Impact of Large Language Models on Programming Education and Student Learning Outcomes," Appl. Sci., vol. 14, no. 10, pp. 4115–4129, May 2024. [Online]. Available: https://doi.org/10.3390/app14104115