Human Factors, Business Management and Society, Vol. 135, 2024, 303–312
https://doi.org/10.54941/ahfe1004957

Artificial Intelligence as a Catalyst: A Case Study on Adaptive Learning in Programming Education

Noora Nieminen[1] and Tero Reunanen[1,2]

[1]Turku University of Applied Sciences, Turku 20520, Finland
[2]University of Vaasa, Vaasa 65200, Finland

ABSTRACT

This paper explores the impact of integrating AI tools in a foundational programming course for first-year higher education students. The AI tools, including generative programming copilot extensions and browser-based AI tools, significantly improved the learning experience. They eased the understanding of basic programming concepts, reduced anxiety, and allowed for more personalized guidance on advanced problems. The AI-enhanced course structure enabled students to tackle more complex programming constructs earlier than traditional curricula. Practical tasks introduced advanced concepts like external data storage, unit testing, and user interface design. Despite incomplete or imperfect projects, students remained motivated and showed resilience in improving their code. The AI's proactive suggestions inspired students to explore beyond the curriculum, delving into databases, cryptography libraries in Python, and advanced user interface design. This experience also encouraged students to learn other programming languages, realizing that individual learning is more accessible with generative AI. In conclusion, AI integration in programming education enhances the learning experience and outcomes, promising to revolutionize traditional teaching methodologies for a more dynamic, responsive, and inclusive learning environment.

Keywords: Artificial intelligence, Education, Programming, Training

INTRODUCTION

In today's tech-driven world, programming education is pivotal in nurturing future innovators. Traditional teaching methods, however, often face issues like student disengagement, limited learning resources, and a gap between theory and practice. AI's advent promises to reshape programming education by offering personalized, interactive learning experiences that bridge the theory-practice gap. This paper explores AI's revolutionary impact in a case study, demonstrating how AI tools address traditional challenges and enrich learning. AI's role in fostering an interactive, personalized, and practical approach to programming education equips students with necessary digital skills. The immense potential of AI in programming education lies in its ability to personalize learning and engage students in real-world applications, unlocking aspiring programmers' full potential. Embracing AI in programming education is crucial for cultivating innovative, skilled programmers for the digital future.

LITERATURE REVIEW

This review discusses the evolution of computer programming education, particularly for first-year higher education students, and the potential integration of AI tools. It highlights the need for diverse teaching strategies (Oliveira et al., 2018), the importance of foundational education in shaping students' future interest in the field (Lockheed, 1986), and addresses key misconceptions in teaching programming to beginners (Qian & Lehman, 2017). It also discusses the balance of theory and practice (Nandigam & Bathula, 2013), and the potential for understanding computing without programming (Urban et al., 2000). The role of programming in problem-solving and computational thinking is underscored (Moon et al., 2020), along with innovative teaching techniques such as visual programming, game-based learning, and collaborative programming (Kanika et al., 2020). The benefits of pair programming (Hanks et al., 2011), the role of creativity (Sharmin, 2021), and the use of digital games (Vahldick et al., 2014) are also discussed The review sets the stage for the integration of AI tools in programming education, exploring the impact and roles of AI teaching assistants. Key factors in adopting AI-assisted education include perceived usefulness and ease of communication (Kim et al., 2020). AI's role in personalizing education and reducing teacher intervention is emphasized (Malik & Solanki, 2021). Despite AI's development in education lagging other fields, interest is increasing (Kurvinen et al., 2022).

Terzopoulos and Satratzemi (2020) have investigated the use of voice assistants and smart speakers in educational settings, emphasizing their potential due to their ease of use and widespread availability. Zhou and Lawless (2015) have discussed AI-assisted smart tutoring systems used in higher education and K-12 settings, highlighting the benefits of AI's round-the-clock availability and its ability to assist students. Zhao and Nazir (2022) have underscored the role of different AI-based applications in enhancing the educational system. They have discussed various applications such as Chatbots, Robotic Assistants, and Vidreader, emphasizing their impact on teaching and learning effectiveness. Kepuska and Bohouta (2018) have discussed the potential of next-generation virtual personal assistants (VPAs) like Microsoft Cortana and Apple Siri in various applications, including education assistance. Cukurova et al. (2019) have presented a case study in debate tutoring, using AI to support educational decision-making processes. They provide empirical evidence for AI's role in enhancing human intelligence augmentation. González et al. (2022) have explored the use of AI Virtual Assistants in software engineering capstone courses, suggesting their potential in enhancing learning experiences through personalized service and collective knowledge leveraging.

The literature review indicates a growing interest and positive perception of AI assistants in education. AI tools are seen as valuable in personalizing learning, enhancing student-teacher interactions, and supporting diverse educational needs. The integration of AI in education represents a significant shift towards more adaptive, responsive, and personalized educational experiences. This context sets the stage for a case study aimed at exploring how first-year higher education students experienced the active use of

AI assistants in a foundational programming course. By integrating AI tools into the course, the study aimed to observe the impact on students' learning curve and overall educational experience. Through this case study, the researchers sought to address existing research gaps by examining AI's potential to enhance the programming learning experience and outcomes. Thus, the literature review supports the statement in the abstract about the importance of gaining student experience in using AI assistants in an introductory programming course.

METHODOLOGY

This study aimed to investigate the impact of integrating AI tools in a foundational programming course for first year higher-education students. In this section, we provide background details of the study, including student prerequisites, course curriculum, and how the course was implemented. We also describe the methods employed to gather data on student experience. Understanding the context and methodology of this study is essential for interpreting the results and drawing meaningful conclusions about the impact of AI on programming education.

Programming Basics course is a compulsory, basic level course for all first-year students in the Bachelor of Engineering, ICT degree program, with a workload of 5 ECTS. The student population in this course consists of degree program students who may be career changers with previous degrees and young adults from secondary schools with varying academic skills. Due to the diverse academic backgrounds of the students, their coding experience and academic competence also vary greatly, which poses a significant challenge in designing a course that can cater to the learning needs of all students. The course's curriculum is structured with a clear objective: to enable students to understand the basics of programming and software application design. Key learning outcomes included understanding software application types, solution technology selection, programming concepts, reading and writing code, decision and control structures, basic object-oriented concepts, and designing simple software applications. The course content covered software application structure, development environment and tools, variables, data types, functions, decision and control structures, and arrays and lists. This course's implementation involved weekly lectures introducing and demonstrating new concepts and theories, followed by practice sessions where students were given practical, lecture-related problems to solve. However, unlike traditional programming courses, where exercises are often repetitive textbook questions, this course focused on open-ended problem-solving tasks, challenging students to think creatively and flexibly with the aid of AI assistants. The tasks were designed to be more real-world related, simulating coding scenarios that students would encounter in their future careers. Additionally, pair and group programming were supported throughout the course, allowing students to collaborate and learn from each other's strengths and weaknesses, rather than solely focusing on individual progress.

The diverse student group comprised 129 enrolled students, with 91 completing the assessment phase. The backgrounds of these students varied

significantly and could be grouped to three distinctive groups: 1) Freshmen with no prior coding experience, 2) Students with some coding knowledge but lack experience designing large-scale programming projects or more advanced algorithmic design skills and 3) Experienced programmers seeking to refresh their skills or learn a new programming language.

The primary challenge was designing a course that catered to this diverse range of backgrounds. Different solutions to tackle this challenge were implemented during the course. Advanced students were allowed to follow an independent online track, allowing them to progress at their own pace. They were provided with individual study materials and resources according to their own interests. Students with some programming knowledge were encouraged to participate in a MOOC alongside the course for additional practice. These students benefit from this auxiliary material by allowing them more flexibility in study pace and providing them a boost in their confidence by letting them individually explore new concepts. Beginners were provided with content and practice tasks designed to be challenging yet manageable, ensuring they were not overwhelmed. All students were encouraged to use AI copilots and AI tutors to assist with weekly assignments and learning new concepts. The tools used included OpenAI's ChatGPT (2023) and Visual Studio Code extensions like GitHub Copilot (2023) and Codeium (2023). These tools were available from the beginning of the course, and students were encouraged to explore and test various features. Critical analysis of the tools' responses and awareness of immaterial and copyright issues related to AI-generated code were emphasized.

To evaluate the impact of these AI tools on learning, students were asked to write a reflection article at the end of the course. This reflection required them to detail their previous experience, how the course affected their understanding and skills in coding, and their perceptions of using AI as a learning tool. The assignment "My Learning Path in the 'Fundamentals of Programming' course" was a reflection essay designed to complement exams and task activities in determining the course grade as a tool for self-assessment. Students were asked to write an essay of 1000–1200 words, discussing their development in programming and self-assessment of their skills. The essay is structured to cover an introduction to their programming background, detailed learning experiences, challenges faced, and their solutions, projects and applications worked on, personal growth in programming perspective, feedback on the course, and future learning interests. It concludes with a summary of the overall experience and learning in the course. This assignment encourages deep reflection on how students have engaged with and understood key concepts, applied them practically, and grown personally and academically through the course. These reflections provided valuable insights into the effectiveness of AI tools in enhancing the programming learning experience for students of varying skill levels and backgrounds.

It is essential to emphasize that this research paper is solely based on qualitative data analysis of reflection articles.75 students returned their reflection articles, and these articles are used as the basis of the qualitative analysis. In this case study, ethical considerations were considered to ensure confidentiality, anonymity, and informed consent. The students were made aware of

the purpose of the data collection, the nature of the questions, and how their responses would be used. The students' identities were kept confidential, and their responses were anonymized during data analysis. The study was conducted in accordance with ethical guidelines and regulations to ensure that the students' rights were respected and protected.

Thematic analysis was used as the primary method for identifying and interpreting patterns within the data. This approach allowed for an in-depth exploration of the students' experiences and perceptions of the AI tools used in the course. To facilitate the analysis, MAXQDA software was utilized. This software enabled efficient organization, coding, and categorization of the large volume of qualitative data. The use of MAXQDA not only streamlined the analytical process but also ensured a systematic and replicable approach to data analysis. The data was coded to correspond student background demographics (beginner, somewhat experienced,advanced), different skills learned during the course (technical/coding, project/group work, growth mindset), attitude towards future programming challenges (I like programming and the course was supporting this attitude, I like programming but this course did not fulfil my expectations,I did not like programming but the course changed my attitude towards programming and me as a learner, I did not know programming but this course made me want to learn more), the use of AI assistants and attitude towards the use of AI (I used and found it beneficial, I used but my experience was negative, I did not use it for ethical / educational reasons).

Through thematic analysis, key themes and insights were extracted from the reflections, providing a comprehensive understanding of how AI tools impacted the students' learning experiences. This methodological approach underscores the validity of the study's findings, offering a nuanced understanding of the integration of AI in programming education. The qualitative nature of this study, underpinned by thematic analysis using MAXQDA, offers valuable insights into the subjective experiences of students, highlighting the nuanced impacts of AI tools in an educational setting.

No quantitative methods have been carried out to ensure that the primary emphasis is on user experience and student reflections. While a qualitative approach is essential in highlighting the potential of AI to transform programming education, it is not without its limitations and biases. One of the main restrictions of this approach is the limited sample size, which may not be representative of the larger population. As such, it is important to exercise caution when interpreting the results of a qualitative study. Additionally, the subjective nature of qualitative data collection may introduce biases in the data, such as the researcher's own biases or the participants' desire to please the researcher. However, despite these limitations, a qualitative approach provides a unique and valuable perspective on the impact of AI tools on programming education. By focusing on student experiences and reflections, we gain valuable insights into how AI tools can transform the traditional teaching methodologies and create a more dynamic, responsive, and inclusive learning environment. This approach highlights the importance of considering the user experience when designing and implementing AI tools in educational settings. Moreover, while quantitative data is typically viewed

as the gold standard in research, a qualitative approach allows for a more in-depth exploration of the complexities and nuances of the impact of AI on programming education. It provides a deeper understanding of the human experience and can uncover unexpected findings that may be missed by quantitative research. Therefore, while acknowledging its limitations, a qualitative approach is essential in complementing quantitative research in highlighting the potential of AI to revolutionize programming education.

RESULTS

We have noticed a significant reduction in the effort required to learn programming since the implementation of the copilot/assistant AI. The AI has acted as an instant helper and tutor to our students, providing assistance with coding problems whenever they need it, even when the teacher or peer students are not available. As a result, our students have found programming to be less daunting and more manageable, leading to a smoother and more enjoyable learning experience. By providing instant support to our students, the assistant AI has become an integral part of our programming classroom and has helped us to create a more supportive and nurturing learning environment. For example, student 4 reflects "*AI has also sometimes been a good help when I have been stuck on a task and simply do not understand what is wrong with my code. When using AI, however, I have had to remember that it cannot do things for you, and one should not blindly trust everything. AI has mainly been helpful in solving individual points of problems, and sometimes it has helped me to look at my own code from a different perspective, thereby guiding me to find the errors myself*" (Student 4, 2023, personal communication) [Translation by author]. Another student delivers a slightly hesitating attitude towards the use of AI, but also recognizes the advantage of obtaining rapid anwers to smaller problems and questions: "*I now have a slightly more positive perspective on using AI for creating frameworks and getting started, kind of like overcoming 'writer's block.' It allows one to hit the ground running in program development, though due to its limitations and perceptions, one has to make an extra round, metaphorically speaking, but at least not reinventing the wheel. There can be different opinions on what is a better approach, but I find it working for me, even though only barely half of the original code and framework suggested by AI remains in principle. I recognize, however, that this was a big challenge for me previously. Another thing is, of course, the so-called stupid questions about details. AI is excellent for that. One wouldn't want to bother real people with them, and it would quickly turn into individual opinions, while the response time on forums, searching, and subjectivity are known issues*" (Student 5, 2023, personal communication) [Translation by author]. Initially, some of our students were hesitant to trust in the code generated by the assistant AI, as seen also in above citation. They were the ones who were most critical of the results given by the AI, but they eventually gave the tool a chance and were positively surprised. Most of our students realized that the code generated by the AI copilot is not always understandable and concise, which made them appreciate the importance of having problem-solving and programming skills to guide the

AI copilot effectively. As a result, our students have gained a deeper understanding of programming concepts and have become more confident in their ability to solve coding problems on their own. The assistant AI has become an essential tool for our students' learning process, helping them to develop their programming skills and become more independent learners. As the reflection articles show, most students mentioned not only technical and programming skills when they were asked to describe the skills they learned – a vast majority of these students mentioned problem solving skills, together with group working skills, project skills and learning growth mindset attitude in learning, meaning that they found the joy of learning more and understanding that they need continuous practice and learning to become experts.

The use of AI has not only made the learning experience smoother, but also allowed us to approach learning new methods from a different perspective. Usually, we would have repetitive exercises to ensure that certain concept has been learned. However, now in the era of assistive AI, this approach can be changed so that these repetitive tasks and exercises can be used in more open-ended problems and larger challenge projects. For example, positive feedback was given especially on the practical and real-world tied weekly practice tasks, which students found more challenging but also more engaging. Many students reported that despite the difficulties, the challenges in problem-solving made them try harder and use different problem-solving approaches, which sometimes were provided by generative AI.

At the end of the course, students in small groups produce code that aims to simulate (in a very simple way using basic data structures, conditional structures, and repetition structures) the operation of an online store. They create a user account, log in, view a product list, add a product to the shopping cart, etc. All this functionality can be implemented using everything already learned, and practical challenges create a need to naturally learn something new. Like in this example, where in this first version users are stored in a list, which disappears when the program closes. Of course, that can't work like that in real life; users should be stored in an external file/database, which can be read and written with new user data. Or how to test the code's functionality without always having to write a main() function to that module and then always execute it from the command line, but instead create a separate test module (basically the fundamentals of unit testing). So, we have already managed in a very short time to start practicing the basic skills of software development at this stage of learning programming. Previously, such things were only considered in object-oriented programming and subsequent courses. But as AI suggests already now, it has encouraged students to explore and understand the use of new things/technologies. The more advanced have been able to progress even further, and they have not become bored while others grapple with basic coding and understanding. Some of the more advanced have started to explore databases on their own, some have started to investigate Python's cryptography libraries (because passwords are no longer stored in plain text in real life!) and some have started to think about how to make a nicer user interface for our 'programs' than the command line.

The course offered more than just technical programming skills. It provided students with real-life programming scenarios that enhanced their

learning experience. With the support of AI, students could explore and understand new technologies. However, some students found that asking open-ended questions from AI copilot did not yield the desired result, which provided them with practical experience on the limitations of AI. By the end of the course, students were able to simulate an online store's operation and practice the basic skills of software development. This hands-on approach not only made the learning process engaging but also prepared students for real-world challenges. Students reported for example the following aspects when asked about their personal growth and progress: their attitude towards programming changed, they not only learned technical skills and coding but also problem solving skills, creativity, project management skills and scheduling. Many students also showed that their attitude towards learning and self-efficacy changed even significantly during the course.

For example Student 1 reflected on their evolving perception of programming, noting, "*My understanding of programming has developed significantly. I have learned to see it as more than just writing code – it is about problem-solving, creative thinking, and continuous learning. The course has also influenced my future goals, and now I am considering delving into Python seriously and starting to implement fun projects*" (Student 1, 2023, personal communication) [Translation by author]. Student 2 reports that "*Project work has been particularly rewarding for me, especially through setting deadlines and achieving them. Through these experiences, I have noticed significant development in both mathematics and programming*"(Student 2, 2023, personal communication). [Translation by author], showing that the skills learned in this basic, AI enhanced programming course delivered transferrable skills beyond programming. Student 3 also noticed that programming is not only coding, stating "*Overall, the 'Fundamentals of Programming' course has been a significant step in my learning path. Although I already had some experience with Python, the course provided a structured and comprehensive approach to the basics of programming. The projects and practical exercises have given me the opportunity to apply what I learned and strengthen my skills. The course has positively affected my perspective on programming, and I have noticed growth both technically and creatively. I have learned that programming is about continuous learning and problem-solving, and I am excited to continue on this learning path*" (Student 3, 2023, personal communication) [Translation by author].

DISCUSSION AND CONCLUSION

Based on the findings of this study, it is important to acknowledge several limitations. The sample size used in this study was relatively small, and therefore the results may not be generalizable to larger populations. Additionally, the duration of the study was relatively short, and future research should consider conducting longitudinal studies to better understand the long-term effects of the intervention being studied. Furthermore, some of the tool-specific factors may have influenced the results. For example, the reliability and validity of the measurement tools used in this study may not have been fully established. Future research should aim to use more reliable and valid measurement tools.

In terms of future research, this study suggests several areas for further investigation. For example, future studies could explore the effectiveness of similar interventions on larger and more diverse populations. Additionally, future research could focus on the long-term effects of the intervention being studied to determine whether the effects are sustained over time. Finally, future research could explore the potential benefits of combining the intervention with other treatments or therapies to enhance its effectiveness.

In conclusion, the study highlights the significant potential of AI in revolutionizing programming education. The key takeaways from the study suggest that AI can assist in personalized learning, provide real-time feedback, improve student engagement, and enhance the overall learning experience. Therefore, it is recommended that educators and institutions consider the integration of AI tools to improve their teaching methodologies and maximize student outcomes. However, it is essential to ensure that the AI tools are used ethically and responsibly, and the educators are adequately trained to implement them effectively. Overall, the integration of AI in programming education can significantly benefit both students and educators, and it is an exciting area to explore for the future of education.

ACKNOWLEDGMENT

The authors wish to express their gratitude for the opportunity to conduct this research in the spirit of open science. We affirm that our work was guided solely by our commitment to contribute to the academic community, free from external demands or influences. Furthermore, we acknowledge that this research was undertaken without the benefit of external funding, underscoring our dedication to the pursuit of knowledge. We hope that our findings will serve to further enrich the discourse in our field.

REFERENCES

Al-Tahat. (2021). "Alice adventures in ComputingLand. A review." 2021 22nd International Arab Conference on Information Technology (ACIT), pp. 1–7. doi: 10.1109/acit53391.2021.9677413.

Codeium. (2023). "Codeium extension for Visual Studio Code." Available at: https://codeium.com/.

Cukurova, M., Kent, C., & Luckin, R. (2019). "Artificial intelligence and multimodal data in the service of human decision-making: A case study in debate tutoring." Br. J. Educ. Technol., 50, pp. 3032–3046.

González, L. A., Neyem, H. A., Contreras-McKay, I., & Molina, D. (2022). "Improving learning experiences in software engineering capstone courses using artificial intelligence virtual assistants." Computer Applications in Engineering Education, 30, pp. 1370–1389.

Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). "Pair programming in education: A literature review." Computer Science Education, 21(2), pp. 135–173. doi: 10.1080/08993408.2011.579808.

GitHub. (2023). "GitHub Copilot." Available at: https://github.com/features/copilot.

Kanika, Chakraverty, S., & Chakraborty, P. (2020). "Tools and techniques for teaching computer programming: A review." Journal of Educational Technology Systems, 49(2), pp. 170–198. doi: 10.1177/0047239520926971.

Kepuska, V., & Bohouta, G. (2018). "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)." 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), pp. 99–103.

Kim, J., Merrill, K., Xu, K., & Sellnow, D. D. (2020). "My teacher is a machine: Understanding students' perceptions of AI teaching assistants in online education." International Journal of Human–Computer Interaction, 36, pp. 1902–1911.

Kurvinen, E., Järvinen, J., & Kaila, E. (2022). "Artificial intelligence in education – Where are we now?" Education and New Developments 2022 – Volume 2.

Lockheed, M. E. (1986). "Trends in Educational Computing: Decreasing Interest and the Changing Focus of Instruction." Educational Researcher, 15(5), pp. 21–26. doi: 10.2307/1174782.

Malik, N., & Solanki, A. (2021). "Simulation of human brain." In Impact of AI Technologies on Teaching, Learning, and Research in Higher Education.

Moon, J., Do, J., Lee, D. et al. (2020). "A conceptual framework for teaching computational thinking in personalized OERs." Smart Learn. Environ. 7, 6. https://doi.org/10.1186/s40561-019-0108-z

Nandigam, D., & Bathula, V. (2013). "Competing Dichotomies in Teaching Computer Programming to Beginner-Students." American Journal of Educational Research, 1(8), pp. 307–312. doi: 10.12691/education-1-8-7.

Oliveira, T., Stringhini, D., & Martins Corrêa, D. G. (2018). "Strategies Focused on the Teaching of Programming Logic: A Systematic Review of Brazilian Literature." 2018 XIII Latin American Conference on Learning Technologies (LACLO), pp. 292–298. doi: 10.1109/LACLO.2018.00059.

OpenAI. (2023). "ChatGPT." Available at: https://openai.com/chatgpt/.

Qian, Y., & Lehman, J. (2017). "Students' Misconceptions and Other Difficulties in Introductory Programming." ACM Transactions on Computing Education (TOCE), 18(1), pp. 1–24. doi: 10.1145/3077618.

Sharmin, S. (2021). "Creativity in CS1: A Literature Review." ACM Transactions on Computing Education (TOCE), 22, pp. 1–26. doi: 10.1145/3459995.

Terzopoulos, G., & Satratzemi, M. (2020). "Voice assistants and smart speakers in everyday life and in education." Informatics Educ., 19, pp. 473–490. Booher, Harold, ed. (2003). Handbook of human systems integration. New Jersey: Wiley.

Urban-Lurain, M., & Weinshank, D. (2000). "Is there a role for programming in non-major computer science courses?" 30th Annual Frontiers in Education Conference. Building on A Century of Progress in Engineering Education. Conference Proceedings (IEEE Cat. No.00CH37135), 1, pp. T2B/7–T2B11 vol. 1, doi: 10.1109/FIE.2000.897590.

Vahldick, A., Mendes, A. J., & Marcelino, M. J. (2014). "A review of games designed to improve introductory computer programming competencies." 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, pp. 1–7. doi: 10.1109/FIE.2014.7044114.

Zhao, Q., & Nazir, S. (2022). "English multimode production and usage by artificial intelligence and online reading for sustaining effectiveness." Mobile Information Systems.

Zhou, M. Y., & Lawless, W. (2015). "An overview of artificial intelligence in education." In Artificial Intelligence: Concepts, Methodologies, Tools, and Applications, pp. 2445–2452.