International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

INTERNATIONAL JOURNAL OF EDUCATION, PSYCHOLOGY AND COUNSELLING (IJEPC)
www.ijepc.com

ASSESSMENT FOR PROGRAMMING SUBJECT IN HIGHER EDUCATION: A STRUCTURED REVIEW

Nor Zalina Ismail[1]*, Mohd Rizal Razak[2], Khairunnisa A.Kadir[3], Rozeleenda Abdul Rahman[4], Mohd Azim Zainal[5]

[1] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (Pahang), Malaysia
Email: nza1601@uitm.edu.my
[2] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (Pahang), Malaysia
Email: dragon_admire007@uitm.edu.my
[3] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (Pahang), Malaysia
Email: khairunnisa.kadir@uitm.edu.my
[4] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (Pahang), Malaysia
Email: rozeleenda@uitm.edu.my
[5] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (Pahang), Malaysia
Email: azim90@uitm.edu.my
* Corresponding Author

Article Info:

Article history:
Received date: 30.06.2025
Revised date: 20.07.2025
Accepted date: 25.08.2025
Published date: 25.09.2025

To cite this document:

Ismail, N. Z., Razak, M. R., Kadir, K. A., Abdul Rahman, R., & Zainal, M. A. (2025). Assessment For Programming Subject In Higher Education: A Structured Review . *International Journal of Education, Psychology and Counseling, 10* (59), 1200-1215.

DOI: 10.35631/IJEPC.1059088

Abstract:

Assessment is quite important for making higher education programming courses effective for both teachers and students. Programming is still a basic skill in computer science and related professions; thus, it's very important to come up with and use the right ways to test students to make sure they do well and that the school is honest. Even though there is more and more research on the subject, we still need to carefully look at how programming tests are set up, graded, and improved using new ideas. The title of this structured systematic literature review (SLR) is "Assessment for Higher Education in Programming Subject: A Structured Review." Its goal is to bring together existing research on how programming education is assessed. Following the PRISMA process, the review was done to make sure it was open and followed strict rules. After a systematic procedure of finding, screening, assessing eligibility, and inclusion, 34 primary studies were chosen from three major academic databases: Web of Science, Scopus, and ERIC. We grouped the findings from the chosen literature into three main areas: (1) Programming Assessment and Evaluation Strategies, which looks at the design, effectiveness, and difficulties of formative and summative assessment models; (2) Innovative Approaches in Programming Education, which focusses on how automation, gamification, and adaptive technologies can be used in assessment design; and (3) Academic Integrity and Student Performance in Programming, which talks about worries about cheating, test anxiety, and fairness in digital assessment settings. The review shows that even if many new and data-driven ways of testing have made

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

students more interested and enhanced their learning, problems like inconsistent feedback, anxiety during automated testing, and cheating still exist. This study adds to the body of knowledge by showing what makes a good assessment framework and pointing out areas where more research is needed in programming education.

Keywords:

Academic, Assessment, Programming

Introduction

In higher education programming classes, assessment is a complex process that is very important for moulding students' learning experiences and outcomes. Good ways to test students not only check their knowledge and skills, but they also get them excited about the subject. A lot of people have employed traditional evaluation methods like multiple-choice and short-answer questions, but these methods don't always show the full range of students' programming skills (Abdalbari & Hafeez, 2019). As computer science education changes, there is a greater need to look into and use a variety of assessment methods that can give a more complete picture of what students can do.

One big problem with evaluating programming classes is that a lot of students fail or drop out of college. People typically blame this problem on the fact that standard assessment methods don't always show how well students can program (Gomes et al., 2016). To solve this problem, teachers have tried out new ways to test students, such as game-based tests, automated testing tools, and peer assessments. These methods are meant to get students more involved, provide them with feedback more quickly, and give a better picture of how well they can program (Gordillo, 2019; Rodríguez-Del-pino et al., 2022; Van Helden et al., 2023). Automated assessment systems, for example, have been demonstrated to boost students' enthusiasm and performance, even though they can occasionally give feedback that students don't understand (Gordillo, 2019).

Also, testing computational thinking (CT) skills, which are an important part of programming instruction, needs to be done in more than one way. Traditional tests that just look at one thing at a time don't often do a good job of measuring the wide range of skills that come with CT. Using both qualitative and quantitative assessment techniques, including question questions, programming tests, and scale surveys, is a better way to get a whole picture of what students can do (Wang et al., 2023). This all-encompassing approach not only fits better with the goals of the school system, but it also gives a more accurate picture of what pupils can do, which helps them learn more effectively.

Moreover, the assessment of computational thinking (CT) skills, which are integral to programming education, requires a multidimensional approach. Traditional single-approach assessments are often insufficient in capturing the diverse competencies associated with CT skills. A more effective strategy involves combining qualitative and quantitative assessment tools, such as question tests, programming tests, and scale surveys, to provide a holistic evaluation of students' abilities (Wang et al., 2023). This comprehensive approach not only

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

aligns better with the educational objectives but also offers a more accurate reflection of students' skills, thereby supporting their learning journey more effectively.

Table 1: Summary of Literature Review

Aspect	Findings
Traditional Assessment Methods	Multiple-choice and short-answer questions are commonly used but may not fully capture programming competencies(Abdalbari & Hafeez, 2019).
Challenges in Programming Assessment	High failure and dropout rates are linked to inadequate assessment methods(Gomes et al., 2016).
Innovative Assessment Techniques	Game-based assessments, automated tools, and peer assessments enhance engagement and provide accurate measures of skills(Gordillo, 2019; Rodríguez-Del-pino et al., 2022; Van Helden et al., 2023).
Automated Assessment Systems	Improve motivation and performance, but may generate difficult-to-understand feedback(Gordillo, 2019).
Assessment of Computational Thinking Skills	Requires a multidimensional approach combining qualitative and quantitative tools for comprehensive evaluation(Wang et al., 2023).

Table 1 illustrates the summary of the literature review in this research paper. The purpose of this structured approach is to understand and implement assessment in higher education programming courses, highlighting the importance of diverse and innovative methods to accurately measure and enhance students' learning outcomes.

As a conclusion, Figure 1 illustrates a concept map highlighting the key dimensions of *Assessment for Higher Education Programming Subject*. The assessment framework is organized into three main areas: Curriculum Development, Evaluation Methods, and Active Learning Strategies. Under curriculum development, elements such as capstone projects, learning outcomes, and project reports emphasize aligning assessment with program objectives. Evaluation methods include both summative and formative assessments, ensuring a balanced approach to measuring student performance. Meanwhile, active learning strategies are reinforced through peer assessment, engagement in research papers, and hands-on techniques that support deeper understanding. Together, these interconnected components underscore the importance of designing comprehensive and effective assessment practices tailored to programming education in higher learning institutions.

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

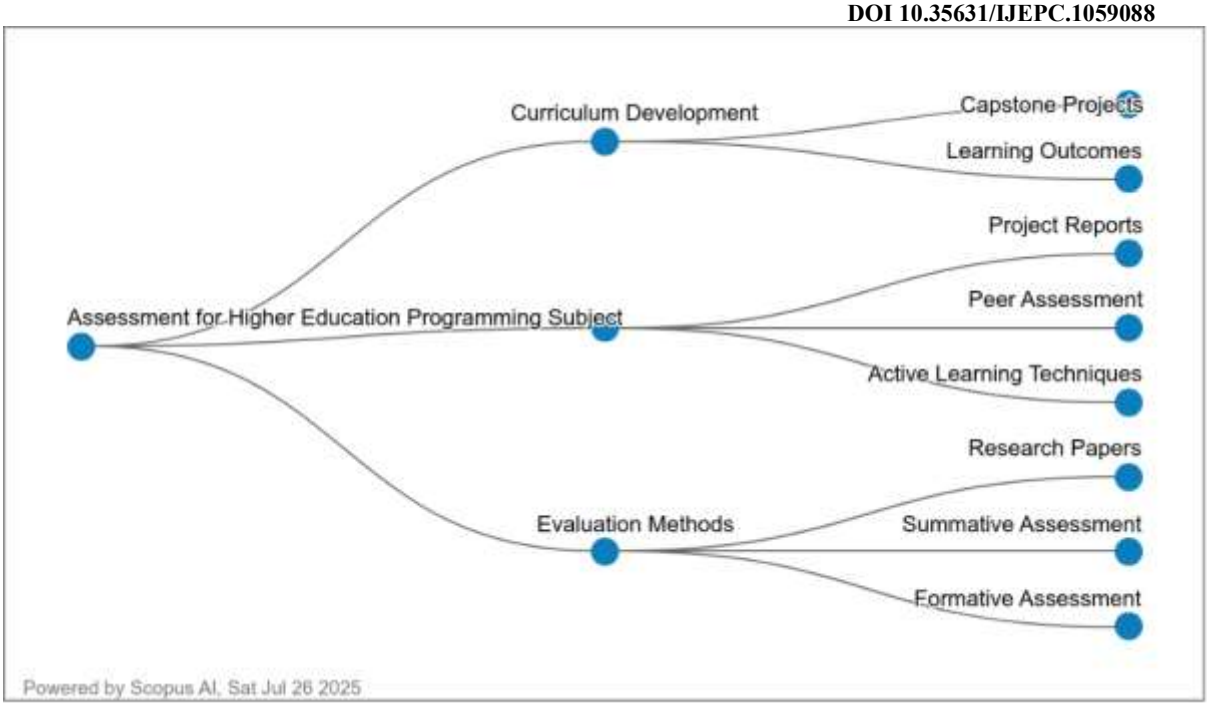

Figure 1: Concept Map for the Introduction Titled "Assessment for Higher Education Programming Subject

(Source: Powered by Scopus AI, Sat, Jul 26 2025)

Literature Review

Assessment in programming subjects within higher education has increasingly shifted toward the use of automated grading and feedback tools. These systems primarily evaluate the correctness of code through dynamic techniques like unit testing and static analysis, providing students with rapid feedback and opportunities for multiple resubmissions. While this approach enhances student satisfaction and reduces instructor workload, it often falls short in assessing code quality aspects such as maintainability, readability, and documentation, which are crucial for real-world programming competence(Messer et al., 2024; Paiva, J., Leal, J., & Figueira, 2022).

To address the need for ongoing evaluation and deeper learning, continuous assessment methodologies have been implemented, often supported by automated tools. These approaches have been shown to improve student motivation, commitment, and performance, as students prefer the flexibility and immediacy of automated assessments over traditional methods. However, the effectiveness of continuous assessment depends on thoughtful design to ensure that it promotes not just frequent testing, but also meaningful engagement and knowledge retention (Calderon, K., Serrano, N., Blanco, C., & Gutierrez, 2023).

Beyond automation, innovative assessment formats such as creative programming projects, serious games, and extracurricular activities are being explored to foster computational thinking, creativity, and engagement. While automated tools can efficiently measure algorithmic complexity and correctness, human assessment remains essential for evaluating creative and higher-order problem-solving skills. Game-based and creative assessments have demonstrated positive effects on learning outcomes and student motivation, suggesting that a balanced combination of automated and human evaluation is necessary for comprehensive

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

assessment in programming education (Arias-Herguedas et al., 2025; Hainey & Baxter, 2024; Romero et al., 2017)

In conclusion, effective assessment in programming education requires a balanced integration of automated and human evaluation methods to address both technical accuracy and higher-order skills. While automated tools offer efficiency, rapid feedback, and support for continuous assessment, they often overlook essential qualitative aspects such as maintainability, readability, and creativity. Incorporating diverse formats like creative projects, serious games, and extracurricular activities enriches learning by fostering computational thinking, engagement, and problem-solving capabilities. A thoughtfully designed assessment strategy that leverages automation for efficiency while preserving human judgment for nuanced evaluation ensures a more comprehensive measure of students' programming competence and prepares them for real-world professional demands (Messer et al., 2024; Paiva et al., 2022; Calderon et al., 2023; Arias-Herguedas et al., 2025; Hainey & Baxter, 2024; Romero et al., 2017).

Material and Methods

The methods used in this systematic review are based on PRISMA (Page et al., 2021), which have the following steps:

Identification

Important phases in the systematic review method were used in this study to collect a significant amount of pertinent material. Choosing keywords was the first step in the procedure. Next, dictionaries, thesauri, encyclopedias, and previous research were used to find similar terms. Search strings for the Web of Science and Scopus databases were constructed when all pertinent terms were found (see Table 2). 772 papers relevant to the study issue were found in the two databases during this first stage of the systematic review.

Table 2: The Search String

Database	Search String
Scopus	TITLE-ABS-KEY (Assessment AND "Programming" AND "Higher Education") AND (LIMIT-TO (DOCTYPE , "ar")) AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (SRCTYPE , "j")) AND (LIMIT-TO (PUBYEAR , 2021) OR LIMIT-TO (PUBYEAR , 2022) OR LIMIT-TO (PUBYEAR , 2023) OR LIMIT-TO (PUBYEAR , 2024) OR LIMIT-TO (PUBYEAR , 2025)) Date of Access: August 2025
Wos	Assessment AND "Programming" AND "Higher Education" (Topic) and 2025 or 2024 or 2023 or 2022 or 2021 (Publication Years) and Article (Document Types) and English (Languages) Date of Access: August 2025

Screening

During the screening phase, research materials that may be relevant are carefully examined to determine their alignment with the established research question(s). This step typically includes selecting studies related to the assessment of programming subjects in higher education. At this

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

point, duplicate entries are also eliminated. Initially, 575 studies were excluded, resulting in 167 remaining papers for further review according to predefined inclusion and exclusion criteria (refer to Table 3). The primary criterion was the type of literature, focusing on sources that offer practical insights, such as reviews, meta-syntheses, meta-analyses, books, book series, chapters, and conference papers that were not addressed in the latest research. The review only considered English-language publications published between 2021 and 2025. A total of 49 publications were removed due to duplication.

Table 3: The Selection Criterion of Searching

Criterion	Inclusion	Exclusion
Language	English	Non-English
Time line	2021 – 2025	< 2021
Literature type	Journal (Article)	Conference, Book, Review
Publication Stage	Final	In Press

Eligibility

In the third stage, referred to as the eligibility phase, 118 articles were shortlisted for detailed evaluation. At this point, each article's title and main content were thoroughly reviewed to confirm their relevance to the study's inclusion criteria and research objectives. As a result, 84 articles were excluded due to reasons such as being outside the research scope, having titles lacking significance, abstracts unrelated to the study's aim, or the absence of full-text access supported by empirical data. This process led to 34 articles being retained for the subsequent review.

Data Abstraction and Analysis

An integrative analysis was used as one of the assessment strategies in this study to examine and synthesise a variety of research designs (quantitative methods). The goal of the comprehensive study was to identify relevant topics and subtopics. The stage of data collection was the first step in the development of the theme. Figure 2 shows how the authors meticulously analysed a compilation of 34 publications for assertions or material relevant to the topics of the current study. The authors then evaluated the current significant studies related to assessment for higher education in the programming subject. The methodology used in all studies, as well as the research results, is being investigated. Next, the author collaborated with other co-authors to develop themes based on the evidence in this study's context. A log was kept throughout the data analysis process to record any analyses, viewpoints, riddles, or other thoughts relevant to the data interpretation. Finally, the authors compared the results to see if there were any inconsistencies in the theme design process. It is worth noting that, if there are any disagreements between the concepts, the authors discuss them amongst themselves.

The authors also compared the findings to resolve any discrepancies in the theme creation process. Note that if any inconsistencies in the themes arose, the authors addressed them with one another. Finally, the developed themes were tweaked to ensure their consistency. To ensure the validity of the problems, we developed three questions as follows:

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

1. How have assessment and evaluation strategies been designed and implemented to measure learning outcomes in higher education programming courses, and what are their impacts on student engagement and academic performance?
2. What innovative tools, pedagogical models, and technologies have been integrated into programming education, and how do they enhance the effectiveness of teaching and learning in higher education settings?
3. What factors influence academic integrity and student performance in programming education, particularly in the context of emerging technologies and diverse assessment environments?

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

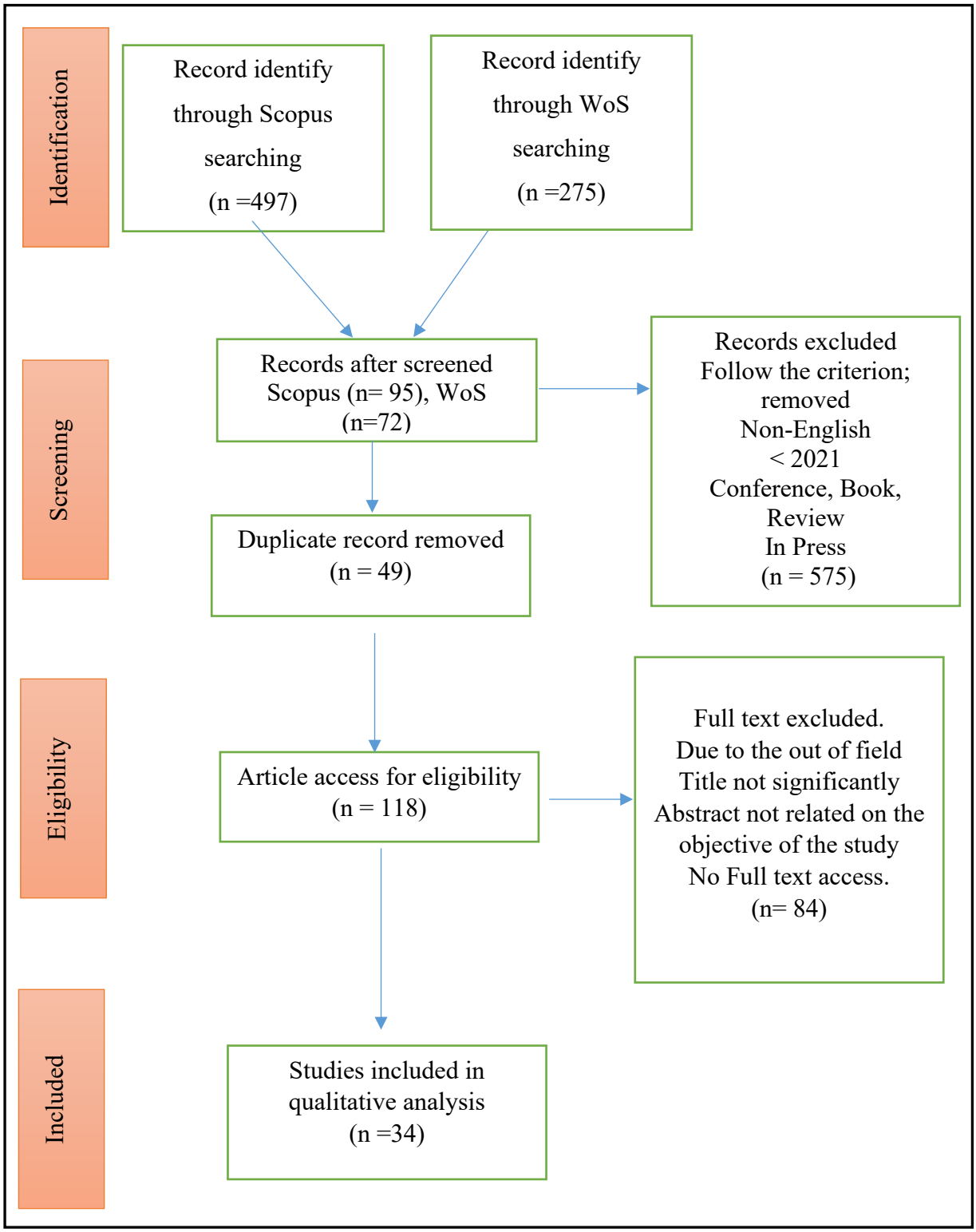

Figure 2. Flow Diagram Of The Proposed Searching Study (Page Et Al., 2021)

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

Results and Discussion

Theme 1: Programming Assessment and Evaluation Strategies

Based on the thematic analysis of the abstracts under the category *"Programming Assessment and Evaluation Strategies"*, several common threads emerge regarding the effectiveness, challenges, and innovations in assessment practices within higher education programming courses. The review below synthesizes these findings with extensive paraphrasing and consolidation across studies, ensuring alignment with academic writing norms and PRISMA-compliant reporting standards.

Several studies emphasized the implementation and challenges of automated and script-based assessment systems. Figueras et al. (2025) explored the dual nature of automated grading systems (AGS), revealing that although AGSs promised efficiency and fair evaluation, their real-world application often disrupted assessment continuity and imposed unexpected burdens on instructors. Similarly, Lapeña-Mañero et al. (2022) introduced an open-source platform for automating non-coding assessment tasks, which demonstrated significant reductions in grading time while enhancing student performance. Modesti (2021) extended this discussion to mobile development, suggesting that script-based assessments not only streamlined grading but also improved students' efficiency in learning technical content. Despite these technological solutions, challenges remained in implementation and in balancing standardization with personalization in evaluating programming performance.

The connection between assessment practices and student learning engagement was another key concern. Veerasamy et al. (2022) examined the use of formative assessments to predict student risk levels. They developed a classification model that linked engagement indicators from ongoing assessments with final performance outcomes, highlighting the value of formative tasks in identifying struggling students early. Tran et al. (2023) utilized data mining techniques to identify gaps between formative and summative assessments, revealing discrepancies in learning topic effectiveness. Their approach helped in continuously improving teaching materials. Sobral (2021), focusing on Bloom’s taxonomy, emphasized how structured cognitive levels can enhance assessment design in introductory programming, guiding both learning objectives and evaluation practices to align with students’ developmental stages.

Another important dimension involved learner diversity and psychological responses to assessment. Tomić et al. (2025) found that automated exams increased anxiety levels among students, particularly females, compared to traditional manual assessments. This anxiety negatively affected their performance, suggesting the need for assessment methods that consider student well-being alongside accuracy and scalability. Riese & Stenbom (2023) observed varied perceptions among engineering students about assessment modes in programming courses. Laboratory tasks were generally welcomed, while exams induced stress, and project tasks were seen as challenging yet authentic. Female students particularly experienced less effective feedback and inconsistency among teaching assistants, raising concerns about equity in assessment experiences. Roque-Hernández et al. (2021) examined pair programming and found that it was positively received by students across gender and experience levels, suggesting collaborative assessment strategies could reduce stress while fostering engagement.

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

Finally, pedagogical adaptability and contextual practices were addressed by Sandstrak et al. (2024), who compared assessment outcomes across campuses applying different instructional models. While assessment formats ranged from practical exams to portfolios, no significant learning outcome differences emerged. This finding reinforced that the choice of assessment format must be contextually appropriate and align with broader instructional strategies rather than being universally applied. In a related context, Ranjeeth & Padayachee (2024) highlighted the influence of internal factors such as problem-solving ability and self-efficacy on programming proficiency. These intrinsic attributes should be considered in designing assessments that accurately capture learner progress and potential.

Theme 2: Innovative Approaches in Programming Education

Based on a comprehensive analysis of the abstract findings within the *Innovative Approaches in Programming Education* theme, several emerging pedagogical strategies and digital interventions have been identified, reflecting significant shifts in how programming is taught at the higher education level. These innovations aim to improve student engagement, self-efficacy, and conceptual understanding in programming courses.

One prominent innovation is the use of game-based and serious game interventions to promote interest and enhance learning outcomes in programming. Zhao et al. (2022) emphasized how serious games contributed to improved knowledge acquisition and increased motivation, particularly when tailored to student demographics. Similarly, Hainey et al. (2022) demonstrated that using games as formative assessment tools improved content retention and test readiness, while Arias-Herguedas et al. (2025) found that integrating games as extracurricular activities resulted in better learning outcomes and higher student motivation across various academic disciplines. The GAME model proposed by Tsai et al. (2024) further validated the role of gamification in enhancing programming self-efficacy and comprehension among non-computer science students, suggesting that game-driven pedagogies are particularly impactful in broadening access to complex computational content.

Beyond gamification, other digital interventions are contributing to personalized and adaptive learning experiences. Sanal Kumar & Thandeeswaran (2025) proposed a rule-based adaptive personalization model for instructional video delivery, enabling individualized pacing based on learner performance and engagement indicators. Similarly, Frialdo et al. (2025) applied augmented reality (AR) in a network systems programming course, observing notable improvements in students' understanding and independent learning through immersive interaction. The PARA application developed by Nannim et al. (2025) focused on robotics programming for preservice teachers and demonstrated the value of project-based, hands-on learning using digital platforms. These findings collectively reinforce the importance of adapting programming instruction to diverse learning styles and technological preferences, offering flexibility and improved cognitive outcomes.

Artificial Intelligence (AI) and automation technologies also play a growing role in enhancing programming education. Dimitrijević et al. (2023) introduced an automated grading framework for Kotlin programming that streamlines assessment and enhances accessibility for mobile development learners. Roldan-Alvarez & Mesa (2024) presented an intelligent deep-learning tutor that provides individualized guidance during programming tasks, reducing the dependency on instructor intervention while enhancing feedback quality. Portella-Cleves & Rodríguez-Hernández (2024) introduced an Active Learning Plan integrating AI tools like

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

ChatGPT, encouraging students to generate, test, and validate pseudocode in collaborative groups, significantly improving readiness for industry-relevant programming challenges. These interventions showcase the potential of AI to transform passive learning into dynamic, feedback-rich experiences.

Collaborative and data-driven approaches also surface as effective strategies for improving programming learning outcomes. Matejic & Milenkovic (2025) highlighted the effectiveness of peer feedback in web programming courses, where students who engaged in feedback cycles outperformed peers without such interactions. This practice enhanced reflection and revision capabilities. Additionally, Chen et al. (2024) developed the Polivr platform to analyze version control system data, identifying learning behaviors and enabling instructors to adjust pedagogical methods in real-time. Such learning analytics frameworks allow for evidence-based instructional decision-making and support early identification of at-risk students, contributing to a more responsive educational environment.

Theme 3: Academic Integrity and Student Performance in Programming

A structured analysis of recent literature under the theme Academic Integrity and Student Performance in Programming reveals an increasing concern over ethical challenges and the evolving role of technology in higher education assessment environments. The findings reflect a multifaceted approach to addressing integrity and performance issues, including technological interventions, collaborative learning, and pedagogical strategies.

One significant strand of the literature explores how emerging technologies—particularly Artificial Intelligence (AI) and automated tools—affect academic integrity in programming courses. Azaiz et al. (2023) found that GPT-3.5 demonstrated potential for providing formative feedback, with accurate assessments in a majority of cases, yet noted limitations in fault localization and error hallucination. Kohen-Vacs et al. (2025) echoed these concerns, indicating that while students perceived generative AI tools as helpful for learning and creativity, they struggled with correcting AI-generated errors during assessments. Humble et al.(2024) emphasized the dual potential of AI tools like ChatGPT: facilitating both enhanced learning and increased opportunities for misconduct, depending on the instructional context. These studies underscore the need to develop students' critical thinking skills in evaluating AI-generated output while balancing efficiency and ethical responsibility.

Another line of inquiry has focused on the development and application of advanced plagiarism detection systems tailored to programming education. Maertens et al. (2022) introduced Dolos, a language-agnostic tool that significantly improves the detection and visualization of code similarity cases, especially in online environments. Karnalim (2023) contributed to this discourse by evaluating SSTRANGE, a similarity detector using locality-sensitive hashing, which outperformed traditional tools in processing efficiency and offered enhanced visualization capabilities. Goldberg (2021), meanwhile, highlighted broader concerns about the fragility of academic integrity frameworks during the pandemic, proposing strategies to reinforce assessments against dishonesty in virtual settings. Together, these works advocate for accessible, robust, and pedagogically informed technological solutions to uphold ethical standards in programming instruction.

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

Collaborative learning and peer-driven approaches have also been investigated for their influence on academic performance and integrity. Matejic & Milenkovic (2025) demonstrated that structured peer feedback activities led to improved outcomes and deeper student reflection in web programming courses. Xu & Correia (2024) validated a new instrument to assess mutual engagement in pair programming, revealing behavioral, cognitive, and emotional components as key to effective collaboration. Schulz et al. (2023) further noted that successful teamwork in programming education requires well-designed collaborative activities, tailored to learning objectives and supported by appropriate instructional scaffolding. These findings suggest that fostering cooperative learning environments not only enhances performance but may also mitigate integrity violations by emphasizing accountability and shared responsibility.

The connection between programming difficulty and student performance has also emerged as a crucial area of investigation. Lokkila et al. (2023) proposed a data-driven model to assess the syntactic difficulty of programming languages, concluding that Python is more accessible for beginners than Java. Their clustering-based evaluation highlighted how language complexity impacts learner performance and stress, which in turn may influence decisions to engage in dishonest behavior. This reinforces the need for adaptive teaching strategies that consider learners' cognitive load, especially in early programming education.

Conclusion

This review highlights the evolving landscape of assessment in higher education programming courses, where traditional methods alone are no longer sufficient to capture the breadth and depth of students' skills. The synthesis of recent studies shows that a balanced blend of formative and summative strategies, supported by technology and grounded in sound pedagogy, can enhance learning outcomes, engagement, and fairness. Innovative approaches—such as gamification, adaptive learning technologies, and AI-assisted feedback—are proving to be valuable tools, while collaborative methods like pair programming and peer review foster both competence and accountability. However, these advancements also bring challenges, including heightened anxiety in automated settings, inconsistent feedback, and the need to uphold academic integrity in an era of rapidly evolving digital tools. Addressing these issues requires context-aware assessment designs that respect student diversity, promote ethical practices, and align with curriculum goals. Ultimately, the findings emphasise that effective programming assessment is not just about measuring knowledge, but about creating an environment that motivates, supports, and prepares students for both academic success and the demands of the professional world.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Acknowledgements

I would like to express my sincere gratitude to the administration of UiTM Cawangan Pahang Kampus Raub for granting me the time off to attend the SLR Seminar. Their support has been invaluable in allowing me to deepen my understanding of the writing skill.

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

References

Abdalbari, A., & Hafeez, K. (2019). Preliminary study on student's assessment in programming courses. *14th International Conference on Computer Science and Education, ICCSE 2019*, 81–84. https://doi.org/10.1109/ICCSE.2019.8845455

Arias-Herguedas, S., Pisabarro-Marron, A., Vivaracho-Pascual, C., Ortega-Arranz, A., & Jiménez, L. I. (2025). Studying the Effectiveness of Games as an Extracurricular Activity in a Higher Education Programming Course. *Computer Applications in Engineering Education*, *33*(2). https://doi.org/10.1002/cae.70000

Azaiz, I., Deckarm, O., & Strickroth, S. (2023). AI-Enhanced Auto-Correction of Programming Exercises: How Effective is GPT-3.5? *International Journal of Engineering Pedagogy*, *13*(8), 67–83. https://doi.org/10.3991/ijep.v13i8.45621

Calderon, K., Serrano, N., Blanco, C., & Gutierrez, I. (2023). Automated and continuous assessment implementation in a programming course. *Computer Applications in Engineering Education*, *32*.

Chen, J., Lau, S., Leinonen, J., Terragni, V., & Giacaman, N. (2024). Detecting Learning Behaviour in Programming Assignments by Analysing Versioned Repositories. *IEEE Access*, *12*(October), 188828–188844. https://doi.org/10.1109/ACCESS.2024.3514843

Dimitrijević, N., Zdravković, N., & Milićević, V. (2023). AN AUTOMATED GRADING FRAMEWORK FOR THE MOBILE DEVELOPMENT PROGRAMMING LANGUAGE KOTLIN. *International Journal for Quality Research*, *17*(2), 313–324. https://doi.org/10.24874/IJQR17.02-01

Figueras, C., Farazouli, A., Cerratto Pargman, T., McGrath, C., & Rossitto, C. (2025). Promises and breakages of automated grading systems: a qualitative study in computer science education. *Education Inquiry*. https://doi.org/10.1080/20004508.2025.2464996

Frialdo, D., Anwar, M., Hendriyani, Y., Sabrina, E., & Hidayat, H. (2025). Enhancing Network Systems Programming Learning through Augmented Reality: A Study on Student Engagement and Understanding. *International Journal of Information and Education Technology*, *15*(4), 774–781. https://doi.org/10.18178/ijiet.2025.15.4.2283

Goldberg, D. M. (2021). Programming in a pandemic: Attaining academic integrity in online coding courses. *Communications of the Association for Information Systems*, *48*, 47–54. https://doi.org/10.17705/1CAIS.04807

Gomes, A., Correia, F. B., & Abreu, P. H. (2016). Types of assessing student-programming knowledge. *Proceedings - Frontiers in Education Conference, FIE*, *2016-Novem*. https://doi.org/10.1109/FIE.2016.7757726

Gordillo, A. (2019). Effect of an instructor-centered tool for automatic assessment of programming assignments on students' perceptions and performance. *Sustainability (Switzerland)*, *11*(20). https://doi.org/10.3390/su11205568

Hainey, T., & Baxter, G. (2024). A Serious game for programming in higher education. *Computers and Education: X Reality*, *4*. https://doi.org/10.1016/j.cexr.2024.100061

Hainey, T., Baxter, G., Black, J., Yorke, K., Bernikas, J., Chrzanowska, N., & McAulay, F. (2022). Serious Games as Innovative Formative Assessment Tools for Programming in Higher Education. *Proceedings of the European Conference on Games-Based Learning*, *2022-Octob*, 253–262. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85141210338&partnerID=40&md5=d73c1eccd2d90e57273996a7a40abeba

Humble, N., Boustedt, J., Holmgren, H., Milutinovic, G., Seipel, S., & Östberg, A.-S. (2024). Cheaters or AI-Enhanced Learners: Consequences of ChatGPT for Programming Education. *Electronic Journal of E-Learning*, *22*(2 Special Issue), 16–29. https://doi.org/10.34190/ejel.21.5.3154

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

Karnalim, O. (2023). Maintaining Academic Integrity in Programming: Locality-Sensitive Hashing and Recommendations. *Education Sciences*, *13*(1). https://doi.org/10.3390/educsci13010054

Kohen-Vacs, D., Usher, M., & Jansen, M. (2025). Integrating Generative AI into Programming Education: Student Perceptions and the Challenge of Correcting AI Errors. *International Journal of Artificial Intelligence in Education*. https://doi.org/10.1007/s40593-025-00496-4

Lapeña-Mañero, P., García-Casuso, C., Montenegro-Cooper, J. M., King, R. W., & Behrens, E. M. (2022). An Open-Source System for Generating and Computer Grading Traditional Non-Coding Assignments. *Electronics (Switzerland)*, *11*(6). https://doi.org/10.3390/electronics11060917

Lokkila, E., Christopoulos, A., & Laakso, M.-J. (2023). A Data-Driven Approach to Compare the Syntactic Difficulty of Programming Languages. *Journal of Information Systems Education*, *34*(1), 84–93. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85148609632&partnerID=40&md5=1659113536909c7cef577d278b0e3cd0

Maertens, R., Van Petegem, C., Strijbol, N., Baeyens, T., Jacobs, A. C., Dawyndt, P., & Mesuere, B. (2022). Dolos: Language-agnostic plagiarism detection in source code. *Journal of Computer Assisted Learning*, *38*(4), 1046–1061. https://doi.org/10.1111/jcal.12662

Matejic, J., & Milenkovic, A. (2025). Impact of Peer Feedback in a Web Programming Course on Students' Achievement. *INTERNATIONAL JOURNAL OF COGNITIVE RESEARCH IN SCIENCE ENGINEERING AND EDUCATION-IJCRSEE*, *13*(1). https://doi.org/10.23947/2334-8496-2025-13-1-33-49

Messer, M., Brown, N. C. C., Kölling, M., & Shi, M. (2024). Automated Grading and Feedback Tools for Programming Education: A Systematic Review. *ACM Transactions on Computing Education*, *24*(1), 1–43. https://doi.org/10.1145/3636515

Modesti, P. (2021). A Script-based Approach for Teaching and Assessing Android Application Development. *ACM Transactions on Computing Education*, *21*(1). https://doi.org/10.1145/3427593

Nannim, F. A., Ibezim, N. E., Agbo, G. C., Mgboji, C., Ngwoke, S. O. R., & Mosia, M. (2025). Development of a Project-Based Arduino Learning App: Fostering Robotics Programming Competence among Preservice Teachers of Computer and Robotics Education. *ACM Transactions on Computing Education*, *25*(1). https://doi.org/10.1145/3719016

Page, M. J., McKenzie, J. E., Bossuyt, P., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., … Moher, D. (2021). The prisma 2020 statement: An updated guideline for reporting systematic reviews. *Medicina Fluminensis*, *57*(4), 444–465. https://doi.org/10.21860/medflum2021_264903

Paiva, J., Leal, J., & Figueira, Á. (2022). Automated Assessment in Computer Science Education: A State-of-the-Art Review. *ACM Transactions on Computing Education (TOCE)*, *22*, 1–40.

Portella-Cleves, J. E., & Rodríguez-Hernández, A. A. (2024). Enhancing Programming Education with an Active Learning Plan and Artificial Intelligence Integration. *REVISTA FACULTAD DE INGENIERIA, UNIVERSIDAD PEDAGOGICA Y TECNOLOGICA DE COLOMBIA*, *33*(67). https://doi.org/10.19053/01211129.v33.n67.2024.16328

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

Ranjeeth, L., & Padayachee, I. (2024). Factors that influence computer programming proficiency in higher education: A case study of Information Technology students. *South African Computer Journal*, *36*(1), 40–75. https://doi.org/10.18489/SACJ.V36I1.18819

Riese, E., & Stenbom, S. (2023). Engineering Students' Experiences of Assessment in Introductory Computer Science Courses. *IEEE Transactions on Education*, *66*(4), 350–359. https://doi.org/10.1109/TE.2023.3238895

Rodríguez-Del-pino, J. C., Hernández-Figueroa, Z. J., Afonso-Suárez, M. D., & González-Domínguez, J. D. (2022). A Comprehensive Discussion of Emerging Automatic Programming Assessment in Learning Management Systems: The VPL Example. In *Microlearning: New Approaches To A More Effective Higher Education* (pp. 141–156). Springer International Publishing. https://doi.org/10.1007/978-3-030-97095-6_9

Roldan-Alvarez, D., & Mesa, F. J. (2024). Intelligent Deep-Learning Tutoring System to Assist Instructors in Programming Courses. *IEEE Transactions on Education*, *67*(1), 153–161. https://doi.org/10.1109/TE.2023.3331055

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, *14*(1). https://doi.org/10.1186/s41239-017-0080-z

Roque-Hernández, R. V, Guerra-Moya, S. A., & Caballero-Rico, F. C. (2021). Acceptance and Assessment in Student Pair-Programming: A Case Study. *International Journal of Emerging Technologies in Learning*, *16*(9), 4–19. https://doi.org/10.3991/ijet.v16i09.18693

Sanal Kumar, T. S., & Thandeeswaran, R. (2025). *An improved adaptive personalization model for instructional video-based e-learning environments*. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85183400710&doi=10.1007%2Fs40692-023-00310-x&partnerID=40&md5=8400974a1c28548be93c3f567cebcceb

Sandstrak, G., Klefstad, B., Styve, A., & Raja, K. (2024). Analyzing Pedagogic Practice and Assessments in a Cross-Campus Programming Course. *IEEE Transactions on Education*, *67*(6), 964–973. https://doi.org/10.1109/TE.2024.3465870

Schulz, S., Berndt, S., & Hawlitschek, A. (2023). Exploring students' and lecturers' views on collaboration and cooperation in computer science courses - a qualitative analysis. *Computer Science Education*, *33*(3), 318–341. https://doi.org/10.1080/08993408.2021.2022361

Sobral, S. R. (2021). Bloom's taxonomy to improve teaching-learning in introduction to programming. *International Journal of Information and Education Technology*, *11*(3), 148–153. https://doi.org/10.18178/ijiet.2021.11.3.1504

Tomić, B., Stojanović, T., Antović, I., & Milić, M. (2025). Students' Test Anxiety and Performance in Introductory Programming: Do Exam and Assessment Modalities Play a Role? *Computer Applications in Engineering Education*, *33*(3). https://doi.org/10.1002/cae.70026

Tran, H., Vu-Van, T., Bang, T., Le, T.-V., Pham, H.-A., & Huynh-Tuong, N. (2023). Data Mining of Formative and Summative Assessments for Improving Teaching Materials towards Adaptive Learning: A Case Study of Programming Courses at the University Level. *Electronics (Switzerland)*, *12*(14). https://doi.org/10.3390/electronics12143135

Tsai, C.-Y., Chen, Y.-A., Hsieh, F.-P., Chuang, M.-H., & Lin, C.-L. (2024). *Effects of a Programming Course Using the GAME Model on Undergraduates' Self-Efficacy and Basic Programming Concepts*. https://www.scopus.com/inward/record.uri?eid=2-s2.0-

International Journal of Education, Psychology and Counseling
IJEPC
EISSN : 0128-164X
Volume 10 Issue 59 (September 2025) PP. 1200-1215
DOI 10.35631/IJEPC.1059088

85174465208&doi=10.1177%2F07356331231206071&partnerID=40&md5=6d8118fd62018890f8406cd22e60f0c3

Van Helden, G., Van Der Werf, V., Saunders-Smits, G. N., & Specht, M. M. (2023). The Use of Digital Peer Assessment in Higher Education-An Umbrella Review of Literature. *IEEE Access*, *11*, 22948–22960. https://doi.org/10.1109/ACCESS.2023.3252914

Veerasamy, A. K., Laakso, M. J., & D'Souza, D. (2022). Formative Assessment Tasks as Indicators of Student Engagement for Predicting At-risk Students in Programming Courses. *INFORMATICS IN EDUCATION*, *21*(2), 375–393. https://doi.org/10.15388/infedu.2022.15

Wang, J., Zhang, W., Zeng, X., & Li, P. (2023). A Computational Thinking Assessment Tool on Text- Based Programming. *2023 IEEE 12th International Conference on Educational and Information Technology, ICEIT 2023*, 326–331. https://doi.org/10.1109/ICEIT57125.2023.10107885

Xu, F., & Correia, A. P. (2024). Measuring mutual engagement in the context of middle-school pair programming: Development and validation of a self-reported questionnaire. *COMPUTERS IN HUMAN BEHAVIOR REPORTS*, *14*. https://doi.org/10.1016/j.chbr.2024.100415

Zhao, D., Muntean, C. H., Chis, A. E., Rozinaj, G., & Muntean, G.-M. (2022). Game-Based Learning: Enhancing Student Experience, Knowledge Gain, and Usability in Higher Education Programming Courses. *IEEE Transactions on Education*, *65*(4), 502–513. https://doi.org/10.1109/TE.2021.3136914