Omeh *et al.*
Journal of New Approaches in Educational Research (2025) 14:22
https://doi.org/10.1007/s44322-025-00041-0

Journal of New Approaches in Educational Research

ORIGINAL RESEARCH Open Access

Fostering programming skill and critical thinking through AI-assisted PBL integration

Christian Basil Omeh[1], Musa Adekunle Ayanwale[2*], Lindelani E. Mnguni[3] and Chijioke Jonathan Olelewe[1]

*Correspondence:
ma.ayanwale@nul.ls

[1] Department of Computer and Robotics Education, University of Nigeria, Enugu, Nigeria
[2] Tests and Measurement Unit, Department of Educational Foundations, Faculty of Education, National University of Lesotho, Maseru 100, Lesotho
[3] Department of Mathematics, Science and Technology Education, Faculty of Education, University of Pretoria, Pretoria, South Africa

Abstract

Despite the increasing emphasis on computational literacy in higher education, we observed that many undergraduate students particularly in developing contexts struggle to master fundamental programming skills and develop critical thinking. Conventional instructional approaches often lack interactivity and personalized scaffolding, which are essential for teaching abstract programming concepts. In response to this challenge, we examined the effect of artificial intelligence (AI)-assisted problem-based learning (PBL) on students' programming skills, critical thinking, and problem-solving abilities in Java programming. Grounded in a quasi-experimental pre-test post-test control group design, we involved 62 s-year computer science education students from two public universities in Nigeria. Participants were assigned to either an experimental group that used Google Gemini (AI) within a PBL framework or a control group exposed to PBL adopting instructional videos. We employed four validated instruments to measure computer programming skills (CPSAR), critical thinking (CTS), problem-solving (PSS), and academic ability. Using multivariate analysis of covariance (MANCOVA), we analyzed group differences while controlling for pre-test scores and tested moderation effects of academic ability and age group. Our results revealed a statistically significant improvement in programming skills among students in the AI-assisted group, with a large effect size. Critical thinking and problem-solving skill outcomes did not differ significantly between the groups. We also found a significant interaction between the teaching strategy and academic ability, indicating that high-ability students benefited more from AI integration into computer programming instruction. This study provides original insights into AI-enhanced pedagogy and has practical implications for improving programming instruction, particularly in resource-limited educational environments.

Keywords: Artificial intelligence in education, Computer programming skills, Critical thinking skills, Problem-based learning

1 Introduction

Computer programming one of the highly sought-after skills in the modern workforce (Maghsoudi, 2024). At the university level, the course on computer application and management course is taught using the Java programming language, which is mandated by the National University Commission (NUC) in Nigerian universities for the

instruction of novice students. However, researchers have established that students' academic achievement and proficiency in computer programming remain limited (Omeh & Olelewe, 2021; Yağcı, 2018). This deficiency hinders students' ability to become self-employed or to function effectively in computing firms. Consequently, this study examines the impact of integrating artificial intelligence technology with Problem-Based Learning (PBL) to explore its efficacy in developing computer programming skills and enhancing critical thinking abilities.

Studies have shown that the poor performance of students in computer programming can be attributed to factors such as the teaching methods adopted by lecturers (Omeh et al., 2022, 2024), inadequate instructional materials (Umakalu, et al., 2025), and large class sizes (Mulryan-Kyne, 2010), among others. Often, students who marginally pass problem-solving courses with poor grades still struggle to understand and translate problems into programming instructions. This challenge is exacerbated by students' varying academic ability levels (Yang, et al., 2024). The author noted that a student's ability to apply what is learned from one context to another is referred to as transferable skills. Researchers continue to seek solutions to improve student academic achievement and skill development (Ateeq, et al. 2024). Abe, et al. (2024) maintained that the admission process for students into computing disciplines should be reviewed, along with the educational curriculum in secondary schools. Furthermore, the theories and models adopted by lecturers and instructors need to be examined (Ladele, et al. 2024), and technologies such as artificial intelligence tools (e.g., Google Gemini) that students interact with daily should be utilized as learning aids.

Researchers have asserted that students with a strong background in mathematics and problem-solving, coupled with higher self-efficacy beliefs, perform better in computer programming courses (Jing, et al., 2024). Giannakos et al. (2025) insisted that adopting a hands-on teaching approach, which allows students to plan their learning activities and determine their learning methods under the guidance of a teacher, will not only improve academic achievement but also enhance skill mastery. Suraworachet, et al. (2024) further asserted that adopting technology that students use daily will enhance learning flexibility. Studies continue to explore the adoption of AI technology in teaching introductory computer programming (Bulathwela, et al., 2024; Viberg et al., 2024). While some studies have shown that AI technology does not support critical thinking, others have found that its adoption enhances critical thinking (Darwin, et al., 2024; Yusuf et al., 2024). This discrepancy in the literature highlights a gap, as the discussion on the adoption of AI in education and its impact on critical thinking remains inconclusive. Therefore, this study combines the PBL approach with AI technology and PBL approach with Video to address this gap. This study examines the impact of teaching computer programming languages using an innovative instructional approach, such as Problem-Based Learning (PBL), introspecting with emerging technologies like Google Gemini and video on enhancing students' critical thinking abilities and their acquisition of computer programming skills.

1.1 Theoretical framework

Stuart Dreyfus (2004) state that individual's progress through different stages of learning and skill acquisition. The five stages of skill acquisition are: novice, advanced beginner,

competent, proficient, and expert. Utilizing Google Gemini and video are considered relevant in creating a conducive learning environment for students in computer programming skills acquisition. Consequently, students are engaged in various task levels: Task 1—problem-solving and algorithm development, Task 2—developing a structured program, Task 3 – program development, Task 4—software application development, and Task 5—comprehensive computer programming tasks. The progression through these tasks is a major factor distinguishing each stage at which a student is situated at a given time. Furthermore, selecting students who are at the same level and class ensures the selection of students with similar background knowledge. This approach aligns with the research question: Does the acquisition of computer programming skills by students taught using PBL-AI and PBL-Video technology improve significantly compared to those taught without it, with respect to computer programming and critical thinking skills?

2 Literature review and hypothesis development

Problem-Based Learning (PBL) is a constructivist, collaborative, contextual, and autonomous learning approach that fosters students' questioning, critical thinking, and problem-solving skills (Wood, 2003). Recent studies have established the positive impact of PBL on students' computing and programming skill development (Omeh et al., 2024; Umakalu et al., 2025). Park and Ertmer (2007) maintain that PBL can be effectively integrated with innovative technologies—such as artificial intelligence (AI), augmented reality, and educational videos—to enhance instructional delivery in technical and computing domains. AI, in particular, has advanced rapidly in recent years, influencing various sectors including education and software development (Fernandes, 2016). The application of AI in programming instruction is increasingly common, with tools such as intelligent tutoring systems, machine learning models, and automated code generation platforms transforming how novice programmers engage with complex content. These tools, including platforms like GitHub Copilot and Google Gemini, provide real-time feedback, error correction, and personalized assistance—features that improve learners' accuracy, reduce cognitive load, and facilitate the development of coding proficiency (Garcia, 2025; Miller, 2004). Despite this progress, there remains a paucity of research specifically examining the integration of AI tools like Google Gemini within PBL frameworks for programming education, particularly in resource-constrained contexts.

In parallel, video technology has emerged as another critical instructional modality. Video-based resources allow learners to observe procedural demonstrations, visualize abstract logic structures, and engage with content at their own pace (Sevin & DeCamp, 2016; Ventura et al., 2015). According to Santagata et al. (2021), students who use video-enriched instructional methods tend to perform better on coding tasks compared to those relying solely on static or text-based resources. Video-supported PBL encourages learner autonomy and fosters conceptual clarity through visual engagement and repetition.

Framing these instructional technologies within a broader theoretical context, Khanchel (2023) offers valuable insight into how technology acceptance, particularly during the COVID-19 pandemic, reshaped learning experiences. Her work underscores that students' engagement with digital platforms—motivated by choice and confidence—was crucial for sustaining educational continuity during lockdowns. This shift

resulted in the formation of virtual communities, where peer interaction, collaborative discourse, and shared inquiry became central to learning. These virtual communities facilitated dynamic knowledge exchange, enhancing learners' absorptive capacity—that is, their ability to comprehend, internalize, and apply new knowledge effectively.

Importantly, such interactions also fostered collective intelligence, wherein the shared insights of participants generated deeper understanding and reflection. Within AI-supported PBL environments, we argue that similar dynamics occur: learners interact not only with peers but also with AI systems, engaging in an iterative process of feedback, exploration, and knowledge construction. The AI functions as a cognitive collaborator, extending learners' capacity to problem-solve, reflect, and adapt in real time. The integration of absorptive capacity and collective intelligence thus offers a robust theoretical lens for understanding how AI-enhanced PBL environments contribute to programming skill development. Taken together, these theoretical and empirical insights support our central proposition that AI-integrated PBL offers a more personalized, interactive, and cognitively enriching instructional experience than video-assisted PBL. Accordingly, we hypothesize that:

H1: There will be a statistically significant difference in post-intervention computer programming skill scores between students taught using PBL-AI (experimental group) and those taught using PBL-Video (control group), after controlling for pre-test scores.

Programming education is more than just learning syntax and algorithms. According to Thorgeirsson and Su (2021) computer programming requires critical thinking and problem-solving skills to develop efficient, logical, and optimized solutions. The authors continues that cognitive abilities enable students to analyze complex problems, troubleshoot errors, and innovate creative solutions. Literature have examined how programming education enhances critical thinking and problem-solving skills, alongside key strategies used in teaching and learning (Parambil, et al. 2024; Zhong & Zhan, 2024; Hu, 2024; Xu, et al., 2023). Critical thinking in programming involves evaluating information, reasoning logically, and making informed decisions. Researchers argue that programming assist in developing f higher-order thinking skills, as students must predict outcomes, assess debugging strategies, and refine algorithms (Kalelioglu & Gülbahar, 2014; Lai & Wong, 2022). Refine algorithm is mostly based on the problem-solving skills of a students. Several instructional methods foster critical thinking and problem-solving abilities in programming education such as pair programming & collaborative coding, gamification & coding challenges and project-based learning (PBL) (Fadilla, et al. 2021). The author recommended a further study on integrating PBL-AI and PBL-Video. Therefore, this study hypothesis that:

H2: There will be a statistically significant difference in post-intervention critical thinking skill scores between the experimental and control groups, controlling for pre-intervention critical thinking skills.

Problem solve is central to programming, requiring students to break down tasks into logical steps, troubleshoot errors, and refine solutions. Studies highlight how

critical thinking divided into decomposition, pattern recognition, abstraction, and algorithmic design plays a crucial role in programming education (Kousar & Afzal, 2021; Masek & Yamin, 2011). By applying iterative debugging techniques, students improve their ability to identify logical errors and enhance efficiency. Researchers advocate for adaptive learning environments that integrate artificial intelligence, interactive tutorials, and real-time feedback to personalize instruction and enhance logical reasoning (Asri, et al. 2024; Ardiansyah et al., 2024). The authors maintained that problem-solving skills are essential components of programming education. By incorporating structured learning techniques, collaborative coding environments and AI-enhanced tutoring, educators can strengthen students' ability to analyze, troubleshoot, and develop efficient programs. Siswanto, (2025) and Yilmaz and Yilmaz, (2023a, 2023b) recommended that future research will focus on immersive learning experiences, such as virtual coding simulations and AI-driven personalized training, to further enhance skill development in programming education.

H3: There will be a statistically significant difference in post-intervention problem-solving skill scores between students in the experimental and control groups, controlling for pre-intervention problem-solving skills.

Computer programming is a fundamental skill in the digital age, enabling individuals to design, develop, and optimize software applications. According to Durak, Yilmaz and Yilmaz, (2019) programming skills encompass various competencies, including problem-solving, logical reasoning, algorithmic thinking, and code efficiency**.** Also, in a study by Polat and Yilmaz (2022) computer programming skills requires proficiency in multiple areas, such as syntax and language proficiency, algorithmic thinking, debugging and optimization. Thus, in software development practices connotes understanding data structures, object-oriented programming, and system architecture to develop effective program instruction. Studies suggest that hands-on learning through projects, coding challenges, and collaborative problem-solving strengthens critical thinking skill (Gulec et al., 2019; Kosa, et al., 2016).

More so, critical thinking is a fundamental cognitive skill that allows individuals to analyze information, evaluate evidence, and make reasoned decisions**.** According to Moraiti, et al (2022) assert that t is crucial in various fields, including education, science, business, and problem-solving scenarios. Researchers argue that fostering critical thinking skills leads to better problem-solving, creativity, and logical reasoning (Basil, 2022; Basil, et al., 2022; Liu, et al., 2022a, 2022b). The authors maintained that critical thinking is essential for navigating complex issues. It further encourages questioning, logical analysis, and evidence-based reasoning helps individuals make informed decisions and solve problems effectively. Basil et al., (2021) in their study found that students with higher problem-solving skills has higher academic standard. Academic ability plays a significant role in learning programming concepts, as students with strong logical reasoning, problem-solving abilities, and mathematical skills often adapt faster to programming (Liu, et al, 2022a, 2022b). There is paucity of research on effect of PBL introspecting with innovative technology (AI technology and video) on computer programming skill, critical thinking skill, and problem-solving skill) moderated by students' academic ability levels. Therefore, we hypothesis that:

H4: The effect of instructional group (PBL-AI vs. PBL-Video) on the combined dependent variables (computer programming skill, critical thinking skill, and problem-solving skill) will be significantly moderated by students' academic ability levels (Low ability vs. High ability).

Programming education is influenced by various factors, including academic ability, cognitive development, critical thinking skill, problem-solving skill and age-related skill acquisition (Doleck, et al., 2017; Li, et al., 2023). The authors continued that students engage with programming at different levels, ranging from basic coding literacy to advanced computational thinking**,** depending on their academic background, intellectual capacity, and age group**.** Moraiti et al. (2022) in their study of how to improve problem solving and critical thinking recommended that age and students' academic ability is a strong factor that influence students learning outcome. According to Hu et al (2016) understanding how these elements shape learning experiences in programming education is crucial for designing effective curricula with respect to students' academic ability. Academic ability plays a significant role in learning programming concepts, as students with strong logical reasoning, problem-solving abilities, and mathematical skills often adapt faster to programming (Ferreira., 2019). Therefore, we hypothesis that:

H6: The three-way interaction between instructional group, students'academic ability, and age group will significantly affect the combined post-intervention outcomes (computer programming skill, critical thinking skill, and problem-solving skill).

Research indicates that students with higher proficiency in analytical thinking excel in coding tasks**,** whereas novice programmers benefit from structured guidance and interactive learning environments (Chuang & Chang, 2024). Programming education is increasingly introduced at younger ages, with platforms such as Scratch and Blockly designed to teach fundamental programming logic to children. Studies suggest that: Young learners (6–12 years) grasp basic computational thinking through visual programming (Rose et al., 2017). Adolescents (13–18 years**)** transition to text-based programming languages, enhancing problem-solving skills (Ghosh, Malva & Singla, 2024). Adults (18 + years) focus on specialized programming domains, applying knowledge to real-world projects and industry applications (Banic & Gamboa, 2019). Therefore, we hypothesis that:

H5: The effect of instructional group on the combined posttest scores will be significantly moderated by student age group (20 years or younger vs. above 20 years).

3 Methodology

3.1 Context, participants, and research design

In this study, we employed a non-equivalent pre-posttest control quasi-experimental research design to examine the effect of Artificial Intelligence (AI)-integrated Problem-Based Learning (PBL) on students' acquisition of computer programming skills, critical thinking and problem-solving abilities. The research was conducted at two public universities in southeastern Nigeria, both offering a core first-year course titled Computer Programming and Management in Schools. This course, taught in Java and mandated by the National Universities Commission (NUC), served as the platform for our experimental design. To preserve natural instructional contexts, we grouped students

based on intact classes, resulting in a sample of 62 students (42 females and 20 males), aged between 16 and 25 years, who had successfully completed prerequisite programming courses (COS 101 and COS 102). These students were divided into an experimental group (PBL-AI, $n=25$) and a control group (PBL-Video, $n=37$). Participants were further categorized by academic ability (low vs. high) and age (20 years or younger vs. above 20 years), creating a factorial framework for analysis. Importantly, our study followed a 2×2x2 factorial structure, comprising three independent variables: instructional strategy (PBL-AI vs. PBL-Video), academic ability (low vs. high), and age group (20 years or younger vs. above 20 years). This factorial arrangement allowed us to assess both main effects and interaction effects among the variables. The matrix formed eight distinct experimental conditions, with participants evenly distributed across them. This approach facilitated rigorous statistical comparisons and ensured the internal validity of our quasi-experimental design.

3.2 Study's treatment package

To evaluate the effectiveness of integrating Artificial Intelligence (AI) technology into computer programming instruction, we designed a six-week intervention involving two groups: an experimental group (PBL-AI) that used Google Gemini as an instructional aid and a control group (PBL-Video) that relied on traditional video-assisted instructional materials. Both groups covered identical curricular content, derived from the National University Commission (NUC) syllabus for Java programming in computer science education. However, the instructional delivery strategies were differentiated, enabling us to isolate the instructional impact of generative AI support within a problem-based learning (PBL) framework.

In Week 1, we commenced the intervention with a joint orientation session for all participants. During this session, we introduced the course objectives, outlined the expectations, and familiarized students with the tools they would use throughout the study. Afterward, we administered the pre-test, comprising validated measures of problem-solving skills, critical thinking, and computer programming skill. Students in the experimental group received a structured introduction to Google Gemini, an AI-based code assistant, emphasizing its role in supporting algorithm generation, code optimization and debugging tasks. In contrast, students in the control group were guided through the use of high-quality Java programming video tutorials curated from academic sources. These videos provided conceptual overviews and illustrated practical programming steps for novice learners (e.g.).

In Week 2, the focus shifted to algorithm design and structured problem-solving. Students in the PBL-AI group explored open-ended real-world problems and leveraged Google Gemini to brainstorm, generate and refine solution algorithms. The students were divided in small groups (e.g. 7-students per group) using the AI assistant to evaluate alternative coding paths and simulate logical flow. During in-class sessions, lecturers and laboratory instructors encouraged metacognitive reflection by prompting learners to compare the algorithmic generated by the Google Gemini AI technology. Conversely, the PBL-Video group was taught by expert-led video content. The videos explained algorithmic design and provided sample problems. Students watched the tutorials, engaged in group discussions, and attempted similar exercises under instructor supervision.

Week 3 was devoted to program structuring and debugging. The PBL-AI group engaged in task-based learning where students wrote Java programs and received AI-generated feedback to identify syntax and logical errors. Google Gemini served as a real-time coach, offering error explanations and suggesting refactored code options. Students iteratively corrected their work based on this feedback, promoting deeper engagement with the debugging process. Meanwhile, the PBL-Video group analyzed prerecorded instructor-led demonstrations of Java programming projects (e.g. projects from laboratory manual). These included identification of common debugging erros, with students practicing replication and solution of similar problems in lab settings.

In Week 4, we emphasized testing and simulation. The PBL-AI group used Google Gemini to test their completed simple Java scripts. Students simulated code execution using logic-based test cases and evaluated output consistency based on AI-guided feedback. This iterative trial-and-error process fostered critical analysis of code structure and output. In contrast, the PBL-Video group held group-based discussion where they collaboratively tested previous sample programs and discussed possible modifications. The technical instructors facilitated the evaluation of code (e.g. in text editor), but without dynamic AI-driven feedback, the students relied primarily on peer explanations and instructor-led clarifications.

Week 5 marked the development of functional software applications. In the PBL-AI group, each student team conceptualized, designed, and implemented a simple Java-based application such as a calculator, student grading system, or login interface. Throughout development, Google Gemini offered support by suggesting logic structures, flagging inefficiencies, and providing user interface enhancement tips. This interactive feedback cycle reinforced self-directed learning and mastery of programming concepts. Students in the PBL-Video group were provided with example prompts from video materials. They developed comparable applications but received instructor feedback after submission rather than real-time assistance.

In Week 6, we focused on consolidation, peer mentoring, and summative evaluation. Students from both groups participated in a peer review exercise, during which they evaluated each other's code based on functionality, clarity, and problem-solving quality. The PBL-AI group utilized Gemini to further revise and polish their solutions before submission. All participants were then re-assessed using the post-test instruments to determine the extent of learning gains. We collected data on each student's post-intervention scores in critical thinking, computer programming, and problem-solving to assess the comparative effectiveness of the two instructional approaches.

3.3 Instrument

To ensure accurate measurement of the key constructs under investigation—academic ability, computer programming skills, critical thinking skills, and problem-solving skills—each instrument was adapted, validated, and tested for reliability. Existing literature, expert reviews, and alignment with our research objectives informed these instruments. Firstly, we adapted an 8-item academic ability test to measure students' reasoning and foundational knowledge in Java programming. The items assessed logical thinking and practical decision-making skills necessary for effective programming. This test served as a moderating variable in the study, distinguishing between students

with high and low academic ability by using the 50th percentile rank to classify student ability levels. Each item presented a scenario-based multiple-choice question with four options, only one correct. For example, a sample item reads: "What should be done when the robot doesn't respond to Java programmed commands?" with possible responses including A. Check for power supply issues and connection errors, B. Ignore the problem and restart the robot, C. Add more commands randomly, D. Update unrelated software. This format ensured cognitive engagement with real-world programming situations. The instrument employed a dichotomous response format (1 = correct, 0 = incorrect), and internal consistency was established using the Kuder-Richardson Formula 20 (KR-20), which yielded a reliability coefficient of 0.79, indicating acceptable internal consistency for group comparison purposes.

We employed the Computer Programming Skills Assessment Rating Scale (CPSAR) to measure students' self-reported computer programming competencies. This instrument consisted of 40 items organized across four functional domains of programming: (1) Problem-solving and algorithm development, (2) Structured programming development, (3) Full program development lifecycle, and (4) Software application development. Each domain was constructed to represent real-world programming tasks. For example, under Task 1: Problem-solving and algorithm development, items included "Proper identification of a problem" and "Good algorithm development." Task 2 included items such as "Ability to use recursion" and "Ability to use application codes' burner for flash." Task 3 covered items such as "Ability to compile/Compilation stage" and "Ability to maintain a program." Task 4 focused on broader software development activities like "Ability to publish a software release" and "Ability to support and add new features to a program." Respondents rated their perceived proficiency on each item using a five-point Likert-type scale, where 5 = Very Good/Perfectly (VG/P) and 1 = Very Poor/Poorly Identified (VP/PI). This scale provided insights into students' confidence and self-perceived competence in handling different aspects of Java programming. The CPSAR was subjected to pilot testing among a group of computer science education students from a comparable institution, resulting in a Cronbach's alpha reliability coefficient of 0.91, indicating excellent internal consistency.

Additionally, we adapted a 7-item instrument based on critical discourse indicators to assess critical thinking during programming activities. This scale evaluated how students demonstrated reflective judgment, argumentation, and evaluation skills within the programming context. Items assessed various aspects of higher-order thinking, such as the ability to justify a viewpoint, analyze other perspectives, and integrate differing ideas into a new perspective. Sample items included: "Do you give substantiated arguments during programming?" "Do you justify your point of view during programming?" and "Do you consider various arguments to form your view?" Each item was rated on a five-point Likert scale ranging from 1 = Strongly Disagree to 5 = Strongly Agree. This scale allowed for a nuanced assessment of students' critical engagement with programming problems and peer contributions. A panel of five experts in science education and educational measurement reviewed and adapted the instrument. Based on Lawshe's (1975) content validation method, items with a Content Validity Ratio (CVR) of 0.78 were retained. The scale's overall Content Validity

Index (CVI) was computed at 0.88, demonstrating excellent content relevance. Pilot testing yielded a Cronbach's alpha of 0.81, confirming acceptable reliability.

Lastly, we used a 6-item scale to evaluate students' problem-solving capacity within Java programming contexts. The items were adapted from validated instruments that assess 21st-century learning and problem-solving skills. The instrument focused on determining how students engage with problems using Java programming tools, how they derive solutions when no immediate path is apparent, and whether they make decisions that align with their goals and confidence. Sample items included: "Does Java programming help you find solutions?" "Do you use Java programming to make informed decisions?" and "Do you solve problems even when solutions are not obvious?" Participants responded using a five-point Likert scale, where 1 = Strongly Disagree and 5 = Strongly Agree. This allowed us to capture individual perceptions of Java programming as a systematic and exploratory problem-solving tool. As with the other instruments, the Problem-Solving Scale underwent validation and reliability testing. Experts evaluated each item using the Lawshe CVR method, and all retained items met the 0.78 threshold. The total CVI for the instrument was 0.84. During pilot testing with a similar cohort, the instrument demonstrated strong internal consistency with a Cronbach's alpha of 0.87.

3.4 Data collection procedure

We conducted the data collection for this study over a structured six-week period, carefully divided into three stages: the preparation phase, the intervention phase, and the assessment phase.

3.5 Preparation phase

We commenced with an orientation session for all participants. During this session, we comprehensively introduced the research objectives, clarified participant roles, and outlined expectations throughout the study. Ethical protocols were adhered to by securing informed consent from each participant, ensuring voluntary participation and confidentiality of responses. After the consent process, we administered the pre-test instruments to all participants across both groups. These instruments included the Computer Programming Skills Assessment Rating Scale (CPSAR), the Critical Thinking Skills Scale (CTS), the Problem-Solving Skills Scale (PSS), and the Academic Ability Test. The pre-test served as a baseline measure of students' competencies prior to the instructional intervention. Participants were also briefed on the nature of the instructional delivery method they would receive either the AI-assisted PBL approach (PBL-AI) or the video-based PBL strategy (PBL-Video). This orientation helped align participant expectations and ensured clarity in instructional goals moving forward.

3.6 Intervention phase

The intervention phase spanned four consecutive weeks, during which students received weekly structured instructional sessions aligned with the treatment package. During this phase, the experimental group engaged in project-based learning supported by Google Gemini, an artificial intelligence tool used for real-time programming assistance, code debugging suggestions, algorithm generation, and logic validation. Each session

encouraged active exploration, where students could query the AI, reflect on its suggestions, and iteratively improve their code. Gemini's role was not to replace cognitive effort but to scaffold thinking, simulate responses, and provoke metacognitive reflection through prompt-based learning. Meanwhile, the control group engaged with curated video-based content that explained similar Java programming concepts. These videos included step-by-step walkthroughs of core programming constructs such as loops, conditional statements, object-oriented principles, and debugging strategies. After watching, students participated in guided discussions with peers and instructors to reflect on the content and apply the learned principles to class-based programming tasks. Both groups completed structured weekly tasks that aligned with the CPSAR domains. For instance, in Week 2, students worked on algorithm design; in Week 3, they focused on debugging tasks; and in Week 4, they developed and tested small software modules. All students documented their progress using programming logs and code journals, which were submitted weekly for formative feedback.

3.7 Post-intervention phase

The final week of the study was dedicated to post-intervention assessment. We re-administered the same set of instruments used during the preparation phase to evaluate changes in students' computer programming skills, critical thinking, and problem-solving abilities. The post-tests were conducted under standardized conditions to reduce measurement error and ensure consistency across both groups. In addition to the post-tests, all students submitted a final programming project that integrated the various learning domains covered during the intervention. These projects were assessed using the CPSAR framework, which allowed us to evaluate not only the students' cognitive development but also their applied competencies in real-world programming scenarios. Project evaluations focused on areas such as algorithm development, code structure, debugging proficiency, documentation, and the overall effectiveness of their software solutions. Throughout the data collection period, we monitored student participation, engagement, and attendance, ensuring that all participants had equitable access to learning resources. The research team promptly addressed any technical or instructional challenges to minimize disruptions. Our data collection procedure allowed us to capture both the immediate and latent effects of the AI-assisted and video-assisted instructional strategies on students' performance and learning progression.

3.8 Method of data analysis procedure

We analyzed the data for this study using IBM SPSS Statistics version 26.0. Our analytical procedure began with thorough data cleaning to ensure the integrity and quality of our dataset. We screened for missing values across all variables. When missing data were detected, we examined patterns to determine whether they were missing completely at random (MCAR), missing at random (MAR), or not missing at random (NMAR). Since the few missing cases were complete at random and constituted less than 5% of the dataset, we addressed them using expectation–maximization (EM) imputation to preserve statistical power and avoid listwise deletion.

Before proceeding to the inferential analyses, we tested the assumptions underlying the Multivariate Analysis of Covariance (MANCOVA). We first assessed

normality by examining the residual distribution using the Shapiro–Wilk and Kolmogorov–Smirnov tests. Although it is common practice to choose either the Shapiro–Wilk or Kolmogorov–Smirnov test based on sample size, we deliberately used both tests to enhance the robustness of our assumption testing, especially considering the sample size in our study ($n = 62$), which falls in the small-to-medium range. As noted by Kim (2013), formal normality tests such as the Shapiro–Wilk and Kolmogorov–Smirnov may be appropriate for samples below 300 but may yield unreliable results with larger datasets. These tests were complemented by visual inspections of Q-Q plots and histograms, which indicated that the residuals were approximately normally distributed across groups.

Next, we tested for homogeneity of variances using Levene's Test for each dependent variable. The test results were not statistically significant, indicating that the assumption of equal variances across groups was met. We also assessed the assumption of homogeneity of variance–covariance matrices using Box's M test. Although this test was statistically significant, suggesting a potential violation of the assumption, we relied on Pillai's Trace as our multivariate test statistic. This choice is supported by the literature, which suggests that Pillai's Trace is robust to violations of homogeneity assumptions and performs well with unequal sample sizes (Van Aelst & Willems, 2011).

After verifying the assumptions, we conducted the MANCOVA to examine whether there were statistically significant multivariate differences in post-test scores across the levels of our independent variables while controlling for the corresponding pre-test scores. The MANCOVA enabled us to assess the joint effect of the independent variables on the set of dependent variables, providing a more comprehensive understanding of the treatment effects. To further explore the multivariate effects and identify which specific dependent variables contributed to the overall significance, we conducted follow-up univariate Analyses of Covariance (ANCOVA) for each outcome. These univariate tests allowed us to determine the unique effect of the independent variables on each dependent variable while still controlling for baseline (pre-test) performance. We also computed estimated marginal means to understand adjusted group means and assist in interpretation. Where significant effects were found, we performed pairwise comparisons with Bonferroni corrections to control for Type I errors due to multiple testing. Finally, to gauge the practical significance of our results, we reported effect sizes using partial eta squared (η^2) for both multivariate and univariate analyses.

3.9 Ethical consideration

We prioritized ethical integrity throughout the study. Approval was obtained from the Faculty of Vocational Technical Education Research Ethics Committee at the University of Nigeria, Nsukka. We ensured voluntary participation by securing informed consent from all students. Participants were briefed on their right to withdraw at any point without consequence. All responses were anonymized, and the dataset was securely stored. We adhered strictly to ethical principles of autonomy, beneficence, and justice, ensuring that the research complied with both institutional and international ethical standards. Our Ref UNN/PG/PhD/2020/95701 on 3/8/2022.

4 Results

4.1 Assessment of underlying assumptions

Before conducting the MANCOVA, we assessed several critical assumptions to ensure the validity of our results. We began by testing the assumption of normality using both the Kolmogorov–Smirnov (KS) and Shapiro–Wilk (WS) tests across the levels of group (experimental vs. control), academic ability (low vs. high), and age category (≤ 20 vs. >20 years). As shown in Table 1, results indicated that the distribution of post computer programming skills (Post_CPS) scores did not significantly deviate from normality in most subgroups. However, there was a slight deviation in the Shapiro–Wilk test among participants aged above 20 years ($p=0.008$), suggesting some skewness in that subgroup. For post problem solving skills (Post_PSS), significant deviations from normality were observed in the experimental group (SW = 0.002; KS = 0.013), and within the high academic ability subgroup (SW = 0.017). The most substantial deviation from normality occurred in post critical thinking skills (Post_CRT), where both the experimental group (SW = 0.000; KS = 0.000) and those aged above 20 (SW = 0.021; KS = 0.002) showed non-normal distributions. Given that the sample sizes in each subgroup exceeded 20 participants, the assumptions of normality for MANOVA can be considered robust to moderate violations (Tabachnick & Fidell, 2007). We further supplemented the formal tests of normality with visual inspections of Q-Q plots. These graphical assessments revealed no severe deviations from normality, indicating that the residuals were approximately normally distributed across the different groups. Additionally, we then tested the assumption of equality of variance–covariance matrices using Box's M Test. The result (see Table 2) was statistically significant, Box's M = 76.342, F(36, 1571.63) = 1.637, $p=0.010$, indicating a violation of this assumption. In response, we relied on Pillai's Trace for interpreting multivariate results, as it is the most robust statistic when

Table 1 Tests of normality for post-test scores by academic ability, age group, and group type

Variable	Category	Kolmogorov–Smirnov (Stat.)	df	Sig	Shapiro–Wilk (Stat.)	df	Sig
Post_CPS	Low Ability	.090	25	.200	.970	25	.639
	High Ability	.123	37	.172	.952	37	.110
	20 Years or Below	.167	22	.114	.931	22	.131
	Above 20 Years	.173	40	.004	.921	40	.008
	Experimental	.128	25	.200	.923	25	.059
	Control	.090	37	.200	.990	37	.979
Post_PSS	Low Ability	.112	25	.200	.980	25	.889
	High Ability	.126	37	.144	.926	37	.017
	20 Years or Below	.134	22	.200	.952	22	.353
	Above 20 Years	.110	40	.200	.950	40	.074
	Experimental	.197	25	.013	.857	25	.002
	Control	.072	37	.200	.968	37	.366
Post_CRT	Low Ability	.142	25	.200	.938	25	.133
	High Ability	.180	37	.004	.868	37	.000
	20 Years or Below	.142	22	.200	.936	22	.165
	Above 20 Years	.181	40	.002	.933	40	.021
	Experimental	.304	25	.000	.762	25	.000
	Control	.107	37	.200	.960	37	.196

Table 2 Box's test of equality of covariance matrices

Box's M	F	df1	df2	Sig
76.342	1.637	36	1571.63	.010

this assumption is violated. Next, we evaluated the Levene's Test for equality of error variances. The test revealed that none of the three dependent variables—Post_CPS, Post_CRT, or Post_PSS—violated this assumption: Post_CPS: $F(7, 54) = 1.610$, $p = .152$; Post_CRT: $F(7, 54) = 1.644$, $p = 0.143$; Post_PSS: $F(7, 54) = 2.092$, $p = 0.060$. As none of the p-values were below 0.05, the assumption of homogeneity of error variance was met across all dependent measures (see Table 3).

4.2 Descriptive statistics for dependent variables by group

We explored the descriptive statistics for each dependent variable across the two groups. The experimental group consistently outperformed the control group in computer programming skills (Post_CPS), scoring a mean of 170.93 (SD = 4.37), while the control group recorded a lower mean of 145.37 (SD = 2.45). For critical thinking skills (Post_CRT), however, the control group showed a marginally higher mean (M = 26.37, SD = 0.66) than the experimental group (M = 24.16, SD = 1.17). A similar trend was observed in problem solving skills (Post_PSS) where the control group had a slight edge (M = 25.90, SD = 0.70) compared to the experimental group (M = 24.94, SD = 1.25), though the difference was minimal (see Table 4). These results suggest that the intervention had a noticeable effect on computer programming skills but limited impact on the other domains.

4.3 Multivariate test assessment

We proceeded to conduct a Multivariate Analysis of Covariance (MANCOVA) to investigate the joint effect of Group (experimental vs. control), Academic Ability, and Age Group on three post-test outcomes Post_CPS, Post_CRT, and Post_PSS while controlling for pre-test scores. The multivariate test revealed a statistically significant main

Table 3 Levene's test of equality of error variances

Dependent Variable	F	df1	df2	Sig
Post_CPS	1.610	7	54	.152
Post_CRT	1.644	7	54	.143
Post_PSS	2.092	7	54	.060

Table 4 Descriptive statistics for dependent variables by group

Variable	Group	M	SD	95% CI
Post_CPS	Experimental	170.93	4.37	[162.16, 179.69]
Post_CPS	Control	145.37	2.45	[140.46, 150.28]
Post_CRT	Experimental	24.16	1.17	[21.81, 26.51]
Post_CRT	Control	26.37	0.66	[25.05, 27.68]
Post_PSS	Experimental	24.94	1.25	[22.44, 27.43]
Post_PSS	Control	25.90	0.70	[24.50, 27.30]

Table 5 Multivariate test results (Pillai's Trace)

Effect	Pillai's Trace	F	df	Error df	Sig	η^2
Group	.434	12.537	3	49	.000	.434
Academic Ability	.042	.716	3	49	.547	.042
Age Group	.061	1.053	3	49	.378	.061
Group × Acad. Abil	.148	2.834	3	49	.048	.148

Table 6 Univariate ANCOVA results by group

DV	F	df	Sig	η^2	Interpretation
Post_CPS	26.05	1	.000	.338	Significant
Post_CRT	2.69	1	.107	.050	Not significant
Post_PSS	0.46	1	.502	.009	Not significant

effect of Group, Pillai's Trace = 0.434, F(3, 49) = 12.537, p <.001, η^2 = 0.434. This suggests that the intervention program significantly affected the combined dependent variables, explaining 43.4% of the multivariate variance. There was also a significant interaction effect between Group and Academic Ability, Pillai's Trace = 0.148, F(3, 49) = 2.834, p = 0.048, η^2 = 0.148, implying that the effectiveness of the intervention varied depending on students' academic ability. However, neither the main effects of Academic Ability (p = 0.547) and Age Group (p = 0.378), nor the interactions of Group × Age (p = 0.531) and Group × Academic Ability × Age (p = 0.774) were significant (see Table 5).

Additionally, we estimated univariate ANCOVA tests to determine which specific dependent variables were affected. The results indicated that Post_CPS was significantly influenced by group membership, F(1, 51) = 26.051, p < 0.001, η^2 = 0.338, confirming a large intervention effect on computer programming skills. However, the group differences for Post_CRT (F = 2.693, p =.107, η^2 =.050) and Post_PSS (F = 0.457, p =.502, η^2 = 0.009) were not statistically significant (see Table 6). These results suggest that while the intervention had a powerful impact on enhancing computer programming skills, it did not significantly affect critical thinking skills or problem-solving skills at the univariate level.

4.4 Pairwise comparisons and estimated marginal means

To further examine group differences following the main MANCOVA and univariate ANCOVA tests, we conducted pairwise comparisons using Bonferroni-adjusted estimated marginal means (see Table 7). This approach was employed to control for Type I error arising from multiple comparisons and to identify where significant differences existed between the experimental and control groups across each dependent variable. For the Computer Programming Skills (Post_CPS) variable, the pairwise comparison revealed a statistically significant difference between the experimental and control groups, p <.001. Specifically, participants in the experimental group scored significantly higher than those in the control group by an average of 25.56 points (Mean Difference = 25.558, SE = 5.007, 95% CI [15.505, 35.611]). This large and significant difference indicates that the use of the intervention (e.g., Google

Table 7 Pairwise comparisons of post-test scores between experimental and control groups

Dependent Variable	Group (I)	Group (J)	Mean Difference (I–J)	Std. Error	Sig. (Bonferroni)	95% Confidence Interval
Post_CPS	Experimental	Control	25.558	5.007	.000	[15.505, 35.611]
	Control	Experimental	−25.558	5.007	.000	[−35.611, −15.505]
Post_CRT	Experimental	Control	−2.206	1.344	.107	[−4.905, 0.493]
	Control	Experimental	2.206	1.344	.107	[−0.493, 4.905]
Post_PSS	Experimental	Control	−0.965	1.428	.502	[−3.831, 1.901]
	Control	Experimental	0.965	1.428	.502	[−1.901, 3.831]

Gemini) had a strong positive effect on participants'programming skills. In contrast, for Critical thinking skills (Post_CRT), no statistically significant difference was found between the experimental and control groups ($p = .107$), although the experimental group showed a higher adjusted mean score (Mean Difference = −2.206, SE = 1.344, 95% CI [−4.905,.493]). While suggestive of a potential trend, the result did not reach the significance threshold. Similarly, for Problem-Solving Skills (Post_PSS), the difference between the groups was not statistically significant ($p = .502$). The mean difference of −0.965 (SE = 1.428, 95% CI [−3.831, 1.901]) indicates that the experimental group performed slightly better, but the result was not robust enough to suggest a definitive advantage due to the intervention. This pattern is further corroborated by the estimated marginal means reported in Table 7, where the adjusted mean for Post_CPS was 170.93 for the experimental group versus 145.37 for the control group, while the margins for Post_CRT and Post_PSS were more comparable.

In continuation of our MANCOVA results, we further examined the estimated marginal means for academic ability and age group (see Fig. 1) to understand the adjusted performance differences on each outcome variable while controlling for the covariates (Pre_CPS, Pre_CRT, and Pre_PSS). We observed higher adjusted means across all three dependent variables (Post_CPS, Post_CRT, and Post_PSS) for students classified as having high academic ability. Specifically: The Post_CPS score for high ability students was higher (M = 160.24) compared to low ability peers (M = 156.05), suggesting that stronger academic performers gained more from the programming instruction. Similarly, Post_CRT scores were higher for high ability students (M = 25.50 vs. 25.02), albeit with a minimal margin. For Post_PSS, a more noticeable difference was observed (M = 26.40 for high ability vs. M = 24.44 for low ability), suggesting that academically stronger students demonstrated better integration of problem-solving skills. Furthermore, students above 20 years demonstrated slightly higher Post_CPS scores (M = 159.60) compared to their younger counterparts (M = 156.69). This could reflect maturity or greater cognitive readiness for abstract programming tasks. Interestingly, Post_CRT scores were higher among students 20 years or younger (M = 26.23) compared to those above 20 (M = 24.30). This suggests that younger learners may have benefitted more from interventions designed to stimulate critical thinking, such as interactive AI technologies. For Post_PSS, the difference was marginal, with students above 20 years showing slightly higher scores (M = 25.79 vs. 25.05), again hinting at potential maturity-related benefits in problem-solving integration. Consequently,

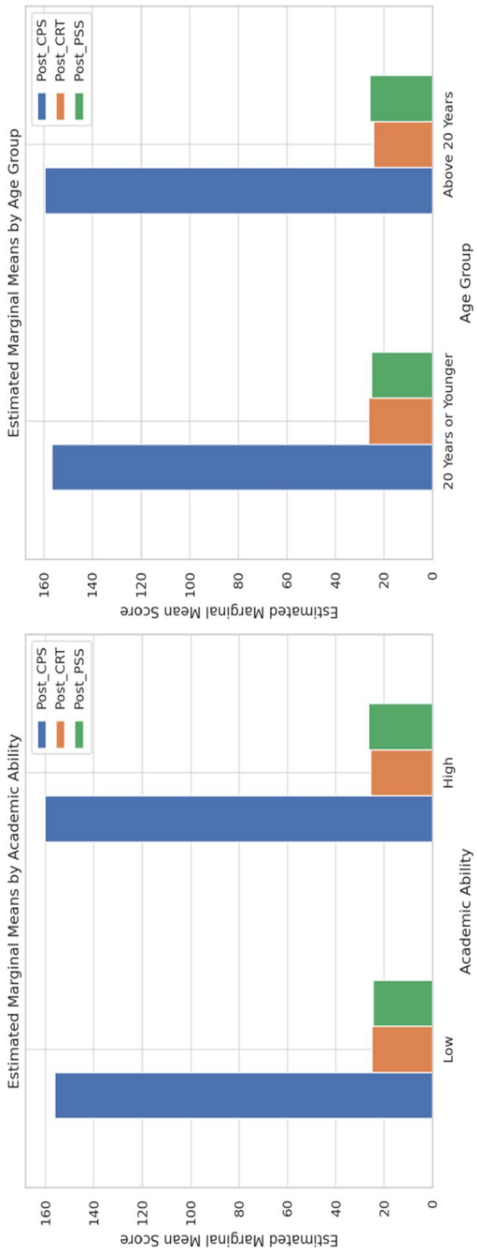

Fig. 1 Estimated marginal means by academic ability and age group

these results not only confirm the significant role of background characteristics (e.g., ability and age) in educational technology research, but also underscore the nuanced interaction between instructional design and learner characteristics. While neither academic ability nor age was statistically significant as a main effect in the MANCOVA model, their estimated marginal means suggest meaningful practical differences that should be explored further in future.

5 Discussion

5.1 Computer programming skill of students taught with PBL-AI and those taught with PBL-Video

The Preliminary analysis such as formal tests of normality, Pillai's Trace for interpreting multivariate results which is the most robust statistic when assumption of normality is violated and Levene's test for equality of error variances was conducted to ensure that assumption of homogeneity of error variance was met across all dependent measures. This is supported by the study of Creswell, et al (2004) who assert that posttest ought to be normal distributed across various subgroup as to ascertain that there is no outliers and the sample size is adequate. The result shown that the intervention PBL-AI group had a statistical significant effect on computer programming skills outcome compare to the students taught with PBL-Video but limited impact on the other domain such as critical thinking skills and problem solving skills. The experimental group fared better than the control group. This finding is consistent with Omeh and Olelewe (2024) who demonstrated that the use of innovative teaching strategies eg. PBL introspecting with learning management system (e.g. Google classroom) had a considerable impact on student academic performance. Herlambang, et al (2024) suggested that students improvement in the academic achievement was attributed to the instructional method (e.g. PBL) coupled with the student's adoption of the emerging technology like AI (Omeh et al. 2025). This clearly suggest that the adoption of PBL-AI as a sound instructional approach that the policy makers in curriculum planning should adopt in the teaching of computer programming skill acquisition. Also the study of Kavitha and Lohani, (2019) contracted with the current study on the adoption of AI technology in enhancing student critical thinking developing.

Test statistics like MANCOVA, univariate ANCOVA and Bonferroni-adjusted estimated marginal means was employed to control for Type I error arising from multiple comparisons. The result also support that there is significant differences between the experimental and control groups in computer programming skills outcome as a result of the intervention (e.g., Google Gemini). This support the findings of Gelman and Tuerlinckx, (2000) maintained that when there are multiple comparison error is likely to occur, but as to ensure that it is established were the statistical significant lies. The result shown that PBL-Video performs better than the PBL-AI group in critical thinking skills and problem-solving skills. This finding is consistent with Lee et al (2025) who demonstrated that the impact of Gen AI in critical thinking reduces cognitive effort. This is shown that the students that adopted PBL-AI did not outperform those that adopted PBL-video. Current study attributed PBL-AI poor development of critical thinking skills and problem-solving skills to over dependent of students on AI during instructional delivery, limited human interaction as seen during the class section and focus on

efficiency over learning. This is supported by Muthmainnah et al (2022) that learners focused more on the how Google Gemini generate the code whenever right prompt was used and the code run in the Java IDE without the students leaning the mastery the learning process. More so study of Melisa et al (2025) maintained that study Adoption of AI does not improve students' development of problem solving especially in higher education. The authors continued that student leaning with AI technology can solve programming problem without requiring user fully understanding the underlying principles. This result into code vibes other than learning how to debug codes and solve the problems.

In contrast both critical thinking and problem-solving skills at posttest shown that there is no statistically significantly different between the PBL-AI and PBL-video. MANCOVA results shown that estimated marginal means for academic ability and age group at posttest shown that student's high academic ability was higher compared to low ability peers, suggesting that stronger academic performers gained more from the programming instruction. This finding is consistent with Liu et al., (2022a, 2022b) who demonstrated a track record of higher academic standards are likely to perform better at another given task reason was a result of adaptability to learning. Sun, et al (2022) and Aru, et al. (2016) also suggested that student who perform in west Africa examination are likely going to perform well in Joint administration and matriculation board suggesting that academically stronger students demonstrated better integration of problem-solving skills.

5.2 PBL-AI vs. PBL-Video on computer programming skill, critical thinking skill, and problem-solving skill and academic ability levels

Multivariate Analysis of Covariance (MANCOVA) was adopted to examine the effect of Group (experimental vs. control), academic ability and age group post-test outcomes of computer programming skill, critical thinking skill, and problem-solving skill. The multivariate test revealed a statistically significant main effect of experimental vs. control. This suggests that the intervention program significantly affected the combined dependent variables, explaining 43.4% of the multivariate variance. The current study shown a significant interaction effect between Group and academic ability. Pillai's Trace implying that the effectiveness of the intervention varied depending on students' academic ability. This is supported by the study of Peng and Kievit, (2020) who posit that student academic ability has always correlated with the student development of skills especially in skill acquisition. This current study sight some factors such mathematical foundation and adaptability to learning as the major influence that result in the students that has higher academic ability to still do well in the development of computer programming skills. This is supported by the findings Costa, et al (2024) who sighted other factors may influence students skills acquisition such as environment and teaching method. This current study shown no significant different on the academic ability and age group nor the interactions of group × age and group × academic ability × age were significant. This supports the findings of Nunes, et al (2014) who assert that age and teaching method do not significant influence skill acquisition especially in higher education. Thus, current study established that teaching method adopted (PBL-AI or PBL-Video) in teaching computer programming skills and age of the student or teaching method adopted

(PBL-AI or PBL-Video) in teaching computer programming skills and academic ability of student has no effect with respect to students' age.

Furthermore, at posttest students above 20 years demonstrated slightly higher scores compared to their younger counterparts. This could reflect maturity or greater cognitive readiness for abstract programming tasks. This is supported by the study of White and Sivitanides (2002) that maintained that the study academic achievement is not a function of student's age. The authors continued that other social variable such parents' social capital, environment and technology contributed to the student s achievement in school. Also, critical thinking development were seen to be higher among students 20 years or younger compared to those above 20. This suggests that younger learners may have benefitted more from interventions designed to stimulate critical thinking, such as interactive AI technologies. This is supported by the study of Psotka, (2013) that younger learners mostly adopt AI technologies like game, virtual reality among others in their study. The author continued that other educational games and virtual reality as disruptive technologies especially in programming. In addition, problem solving skills development shown a marginal difference with students above 20 years showing slightly higher scores, again hinting at potential maturity-related benefits in problem-solving integration. This Abrar, et al (2025) is supported by their study on young learners navigating Problem-Solving with AI technology opined that problem solving skills development is usually higher as a result of the learners adopt of AI tools. Consequently, these results not only confirm the significant role of background characteristics (e.g., ability and age) in educational technology research, but also underscore the nuanced interaction between instructional design and learner characteristics. While neither academic ability nor age was statistically significant as a main effect in the MANCOVA model, their estimated marginal means suggest meaningful practical differences that should be explored further in future.

6 Implications for practice and policy

Our study offers several practical and policy-relevant implications for enhancing programming education, particularly within resource-constrained educational systems. From a practical standpoint, the integration of AI tools like Google Gemini into problem-based learning (PBL) environments presents a promising instructional strategy for improving students' programming skills. We demonstrated that AI-assisted instruction can serve as a real-time scaffold, providing personalized feedback and supporting learners in developing and debugging code. Educators and curriculum designers should consider embedding AI tools into programming courses to facilitate adaptive learning, foster engagement, and close competence gaps, especially among novice programmers. For classroom practice, teacher training programs must incorporate digital pedagogies that build both technological proficiency and instructional confidence. Equipping instructors with the skills to guide students in AI-supported environments will enhance instructional effectiveness and ensure meaningful learner-tool interaction. At the policy level, our findings support the integration of AI-driven learning environments into national curricula for computer science and STEM education. Ministries of Education and higher education regulatory bodies should consider formulating policy frameworks that promote AI literacy, ensure equitable access to AI tools, and guide ethical AI usage

in instructional settings. Furthermore, investment in digital infrastructure is crucial to ensure that AI-supported interventions are scalable across diverse educational contexts. By bridging the gap between pedagogy and technology, our study underscores the value of innovation-driven education policy. Targeted interventions like AI-assisted PBL can contribute to achieving national goals for digital competence, equity, and employability in 21st-century learning environments.

7 Conclusion

The aim of the current study was to empirically ascertain how programming skill and critical thinking skills is fostered through AI-assisted PBL integration (PBL-AI vs PBL-Video) in developing country. The result shown that the adoption of the PBL-AI and PBL-Video improved student acquisition of computer programming outcome. Despite the fact that computer programming skills outcome is crucial in today's workplace, research indicates that it will soon be a required ability for employment especially in AI driven world. Thus, it is important that students develop computer programming skills using innovative pedagogy such as PBL-AI and PBL-video. The result shown that the intervention PBL-AI group had a statistically significant effect on computer programming skills outcome compare to the students taught with PBL-Video but limited impact on the other domain such as critical thinking skills and problem-solving skills. The result shown that students that adopted PBL-Video performs better than the PBL-AI group in critical thinking skills and problem-solving skills. Both critical thinking and problem-solving skills at posttest shown that there is no statistically significantly different between the PBL-AI and PBL-video. MANCOVA results shown that estimated marginal means for academic ability and age group at posttest shown that student's high academic ability was higher compared to low ability peers, suggesting that stronger academic performers gained more from the programming instruction. The current study shown a significant interaction effect between group and academic ability. Also, it was shown that there is no significant different on the academic ability and age group nor the interactions of group × age and group × academic ability × age were significant. Furthermore, at posttest students above 20 years demonstrated slightly higher scores compared to their younger counterparts. This could reflect maturity or greater cognitive readiness for abstract programming tasks. Also, critical thinking development were seen to be higher among students 20 years or younger compared to those above 20. This suggests that younger learners may have benefitted more from interventions designed to stimulate critical thinking, such as interactive. Therefore, university lecturers and technologists teaching computer programming are encouraged to apply innovative teaching practices eg PBL introspecting with artificial intelligent (Google Gemni) and Video for the effective development of computer programming skills in the learners.

8 Limitations and future directions

While our study offers valuable insights into the integration of artificial intelligence in programming instruction, several limitations warrant consideration. First, we employed a non-randomization trial and a quasi-experimental research design, which may limit the generalizability of findings due to potential pre-existing differences

among participants thus, using a randomization trial, pure quasi-experiment research can enhance the study. Although we statistically controlled for pre-test scores, random assignment was not feasible, and unmeasured confounding variables may have influenced the outcomes. Second, the intervention spanned only six weeks, which may not have been sufficient to yield substantial gains in higher-order cognitive outcomes such as critical thinking and problem-solving. These constructs often require prolonged exposure to complex tasks, iterative reflection, and deeper scaffolding that may extend beyond the temporal scope of our study. Additionally, the reliance on self-report instruments for some constructs could introduce response bias, despite our efforts to validate the instruments. Third, our study was context-specific (Java Programming), focusing on computer science education students from two Nigerian universities, mar the generation of the study findings to another domain. As such, cultural and infrastructural factors such as access to reliable internet and exposure to AI tools may have influenced students' engagement with the intervention and should be considered when interpreting the results. Future research should explore longitudinal designs that investigate the sustained impact of AI-assisted PBL on critical thinking and problem-solving skills. Experimental studies with randomized designs and multi-institutional samples are also recommended. Furthermore, qualitative investigations into students'cognitive processes during AI interactions may offer deeper insights into how AI tools mediate learning and foster higher-order thinking.

Acknowledgements
We are grateful to the second-year students from the participating universities who actively engaged with the intervention and provided invaluable feedback. We appreciate the support of faculty members from the Departments of Computer and Robotics Education at the University of Nigeria and the Centre for Teaching and Learning for facilitating logistical arrangements. Special thanks to the research assistants who helped monitor the implementation of the treatment and to our institutional colleagues who reviewed earlier drafts of the instruments and framework.

Authors' contributions
C.B. and M.A. jointly conceptualized the study and developed the instructional design. C.B. led the data collection process, while M.A. oversaw the statistical analysis and interpretation of the findings. L.E. provided expert input on the pedagogical framing of AI in PBL contexts and critically reviewed the manuscript for important intellectual content. C.J. contributed to the adaptation of the instrument and reviewed the implementation of the intervention. All authors contributed to writing the manuscript, critically revised the content, and approved the final version for publication.

Funding
This study did not receive any external funding from public, commercial, or not-for-profit organizations.

Data availability
The datasets generated and analyzed during this study are not publicly available in order to maintain participant confidentiality; however, they are available from the corresponding author upon reasonable request.
The datasets used and analyzed during this study are available from the first author upon reasonable request.

Declarations

Ethics approval and consent to participate
This study was conducted in accordance with established ethical principles for research involving human participants. Ethical approval was obtained from the Faculty of Vocational and Technical Education Research Ethics Committee at the University of Nigeria, Nsukka, under approval number FED/VTE/ETH/2024/04. Prior to data collection, we obtained written informed consent from all participants. Students were informed about the research objectives, data confidentiality protocols, and their rights, including voluntary participation and the freedom to withdraw from the study at any time without penalty. All collected data were anonymized and stored securely.

Consent for publication
All authors have reviewed the final manuscript and consented to its submission and publication in its current form.

Competing interests
The authors declare that there are no financial, professional, or personal competing interests that could have influenced the outcomes of this research.

Received: 19 April 2025 Accepted: 23 July 2025
Published online: 30 September 2025

References

Abe, O. O., Lawal, O. O., Olumide, A. T., & Timilehin, A. (2024). Admission screening mechanism, eliminating distance obstacles. In: *2024 IEEE 5th International Conference on Electro-Computing Technologies for Humanity (NIGERCON)* (pp. 1–5). IEEE.

Abrar, F., Mehmood, S., & Kandhro, A. N. (2025). Cognitive development and AI: A longitudinal study of children and adults navigating problem-solving with AI tools. *The Critical Review of Social Sciences Studies, 3*(1), 1888–1904.

Ardiansyah, A. I., Putra, A. K., & Nikitina, N. (2024). Investigating problem-based learning model's impact on student's critical thinking skills in environmental conservation context. *Jambura Geo Education Journal, 5*(2), 87–103.

Aru, O. E., Achumba, I. E., & Opara, F. K. (2016). Assessment of the admission criteria that predict students' academic performance in undergraduate years in a Nigerian University. In: *Proceedings on the International Conference on Artificial Intelligence (ICAI)* (p. 356). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Asri, I. H., Jampel, I. N., Arnyana, I. B. P., Suastra, I. W., & Nitiasih, P. K. (2024). Profile of Problem Based Learning (PBL) model in improving students' problem solving and critical thinking ability. *KnE social sciences*, 769–778.

Ateeq, A., Alaghbari, M. A., Alzoraiki, M., Milhem, M., & Beshr, B. A. H. (2024). Empowering academic success: integrating AI tools in university teaching for enhanced assignment and thesis guidance. In: *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)* (pp. 297–301). IEEE.

Banic, A., & Gamboa, R. (2019). Visual design problem-based learning in a virtual environment improves computational thinking and programming knowledge. In: *2019 IEEE Conference on virtual reality and 3D User Interfaces (VR)* (pp. 1588–1593). IEEE.

Basil, O. C. (2022). Cloud computing and undergraduate researches in universities in Enugu State: Implication for skills demand. *International Journal of Instructional Technology and Educational Studies, 3*(2), 27–33.

Basil, O. C., Nwokoye, E., & Aduku, E. B. (2021). Computing education, decent work and economic growth in Nigeria. *International Journal of Economics Development Research (IJEDR), 2*(1), 44–64.

Basil, O., Umakalu, C., & Nwangwu, E. (2022). Effect of Google Classroom on academic achievement of undergraduate students in computer database management system in universities in south east Nigeria. *International Journal of Instructional Technology and Educational Studies, 3*(1), 9–15.

Bulathwela, S., Pérez-Ortiz, M., Holloway, C., Cukurova, M., & Shawe-Taylor, J. (2024). Artificial intelligence alone will not democratise education: On educational inequality, techno-solutionism and inclusive tools. *Sustainability, 16*(2), 781.

Chuang, Y. T., & Chang, H. Y. (2024). Analyzing novice and competent programmers' problem-solving behaviors using an automated evaluation system. *Science of Computer Programming, 237*, Article 103138.

Costa, A., Moreira, D., Casanova, J., Azevedo, Â., Gonçalves, A., Oliveira, Í., Azevedo, R., & Dias, P. C. (2024). Determinants of academic achievement from the middle to secondary school education: A systematic review. *Social Psychology of Education, 27*(6), 3533–3572.

Creswell, J. W., Fetters, M. D., & Ivankova, N. V. (2004). Designing a mixed methods study in primary care. *The Annals of Family Medicine, 2*(1), 7–12.

Darwin, Rusdin, D., Mukminatien, N., Suryati, N., Laksmi, E. D., & Marzuki. (2024). Critical thinking in the AI era: An exploration of EFL students' perceptions, benefits, and limitations. *Cogent Education*, *11*(1), 2290342.

Doleck, T., Bazelais, P., Lemay, D. J., Saxena, A., & Basnet, R. B. (2017). Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving: Exploring the relationship between computational thinking skills and academic performance. *Journal of Computers in Education, 4*, 355–369.

Dreyfus, S. E. (2004). The five-stage model of adult skill acquisition. *Bulletin of Science, Technology & Society, 24*(3), 177–181.

Durak, H. Y., Yilmaz, F. G. K., & Yilmaz, R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology, 10*(2), 173–197.

Fadilla, N., Nurlaela, L., Rijanto, T., Ariyanto, S. R., Rahmah, L., & Huda, S. (2021). Effect of problem-based learning on critical thinking skills. *Journal of Physics: Conference Series, 1810*(1), 60. IOP Publishing.

Fernandes, M. A. (2016). Problem-based learning applied to the artificial intelligence course. *Computer Applications in Engineering Education, 24*(3), 388–399.

Ferreira, B. M., de Souza, L. M., Silva, L. das A., Felix, I. M., Brandão, L. de O., & Brandão, A. A. F. (2019). The Marriage of Mathematics and Programming. https://doi.org/10.5753/CBIE.SBIE.2019.1790

Garcia, M. B. (2025). Teaching and learning computer programming using ChatGPT: A rapid review of literature amid the rise of generative AI technologies. *Education and information technologies*, 1–25.

Gelman, A., & Tuerlinckx, F. (2000). Type s error rates for classical and Bayesian single and multiple comparison procedures. *Computational Statistics, 15*(3), 373–390.

Ghosh, A., Malva, L., & Singla, A. (2024). Analyzing-evaluating-creating: Assessing computational thinking and problem solving in visual programming domains. In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* (pp. 387–393).

Giannakos, M., Horn, M., & Cukurova, M. (2025). Learning, design and technology in the age of AI. *Behaviour & Information Technology, 44*(5), 883–887.

Gulec, U., Yilmaz, M., Yalcin, A. D., O'Connor, R. V., & Clarke, P. M. (2019). Cengo: A web-based serious game to increase the programming knowledge levels of computer engineering students. In: *Systems, Software and Services Process Improvement: 26th European Conference, EuroSPI 2019, Edinburgh, UK, September 18–20, 2019, Proceedings 26* (pp. 237–248). Springer International Publishing.

Herlambang, A. D., Ramadana, M. R., Wijoyo, S. H., & Phadung, M. (2024). Students' cognitive load on computer programming instructional process using example-problem-based learning and problem-based learning instructional model at vocational high school. *Elinvo (Electronics, Informatics, and Vocational Education), 9*(2), 309–320.
Hu, L. (2024). Programming and 21st century skill development in K-12 schools: A multidimensional meta-analysis. *Journal of Computer Assisted Learning, 40*(2), 610–636.
Hu, W., Jia, X., Plucker, J. A., & Shan, X. (2016). Effects of a critical thinking skills program on the learning motivation of primary school students. *Roeper Review, 38*(2), 70–83.
Jing, Y., Wang, H., Chen, X., & Wang, C. (2024). What factors will affect the effectiveness of using ChatGPT to solve programming problems? A quasi-experimental study. *Humanities and Social Sciences Communications, 11*(1), 1–12.
Kalelioglu, F., & Gülbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33–50.
Kavitha, V., & Lohani, R. (2019). A critical study on the use of artificial intelligence, e-learning technology and tools to enhance the learners experience. *Cluster Computing, 22*(Suppl 3), 6985–6989.
Khanchel, H. (2023). Factors affecting social network use by students in Tunisia. *Human Systems Management, 42*(2), 131–148. https://doi.org/10.3233/HSM-220017
Kim, H. Y. (2013). Statistical notes for clinical researchers: Assessing normal distribution (2) using skewness and kurtosis. *Restorative Dentistry & Endodontics, 38*(1), 52. https://doi.org/10.5395/rde.2013.38.1.52
Kosa, M., Yilmaz, M., O'Connor, R., & Clarke, P. (2016). Software engineering education and games: A systematic literature review. *Journal of Universal Computer Science, 22*(12), 1558–1574.
Kousar, R., & Afzal, M. (2021). The effects of problem based learning on critical thinking and problem solving skills among midwifery students. *Pakistan Journal of Medical and Health SciencEs, 15*(4), 722–725.
Ladele, O. A., Abubakar, R. B., Ariba, O. T., spsampsps Ajani, T. O. (2024). A Systematic review of the training of mathematics teachers in Nigeria (1990–2020). *Mathematics teacher training and development in Africa: Trends at primary and secondary school levels*, 131–147.
Lai, X., & Wong, G. K. W. (2022). Collaborative versus individual problem solving in computational thinking through programming: A meta-analysis. *British Journal of Educational Technology, 53*(1), 150–170.
Lawshe, C. H. (1975). A quantitative approach to content validity. *Personnel Psychology, 28*, 563–575. https://doi.org/10.1111/j.1744-6570.1975.tb01393.x
Lee, H. P. H., Sarkar, A., Tankelevitch, L., Drosos, I., Rintel, S., Banks, R., & Wilson, N. (2025). *The impact of generative AI on critical thinking: Self-reported reductions in cognitive effort and confidence effects from a survey of knowledge workers*.
Li, W., Liu, C. Y., & Tseng, J. C. (2023). Effects of the interaction between metacognition teaching and students' learning achievement on students' computational thinking, critical thinking, and metacognition in collaborative programming learning. *Education and Information Technologies, 28*(10), 12919–12943.
Liu, F., Zhao, L., Zhao, J., Dai, Q., Fan, C., & Shen, J. (2022a). Educational process mining for discovering students' problem-solving ability in computer programming education. *IEEE Transactions on Learning Technologies, 15*(6), 709–719.
Liu, H., Sheng, J., & Zhao, L. (2022b). Innovation of teaching tools during robot programming learning to promote middle school students' critical thinking. *Sustainability, 14*(11), 6625.
Maghsoudi, M. (2024). Uncovering the skillsets required in computer science jobs using social network analysis. *Education and Information Technologies, 29*(10), 12759–12780.
Masek, A., & Yamin, S. (2011). The effect of problem based learning on critical thinking ability: A theoretical and empirical review. *International Review of Social Sciences and Humanities, 2*(1), 215–221.
Melisa, R., Ashadi, A., Triastuti, A., Hidayati, S., Salido, A., Ero, P. E. L., et al. (2025). Critical thinking in the age of AI: A systematic review of AI's effects on higher education. *Educational Process: International Journal*.
Miller, D. P. (2004). Using robotics to teach computer programming & AI concepts to engineering students. In: *Proceedings of the AAAI Spring Symposium on Accessible Hands-on Artificial Intelligence and Robotics Education*.
Moraiti, I., Fotoglou, A., & Drigas, A. (2022). Coding with block programming languages in educational robotics and mobiles, improve problem solving, creativity & critical thinking skills. *International Journal of Interactive Mobile Technologies*. https://doi.org/10.3991/ijim.v16i20.34247
Mulryan-Kyne, C. (2010). Teaching large classes at college and university level: Challenges and opportunities. *Teaching in Higher Education, 15*(2), 175–185.
Muthmainnah, Ibna Seraj, P. M., & Oteir, I. (2022). Playing with AI to investigate human-computer interaction technology and improving critical thinking skills to pursue 21st century age. *Education Research International*, *2022*(1), 6468995.
Nunes, M. E., Souza, M. G., Basso, L., Monteiro, C. B., Corrêa, U. C., & Santos, S. (2014). Frequency of provision of knowledge of performance on skill acquisition in older persons. *Frontiers in Psychology, 5*, 1454.
Omeh, C. B., & Olelewe, C. J. (2021). Assessing the effectiveness of innovative pedagogy and lecture method on students academic achievement and retention in computer programming. *Education Research International, 2021*(1), 5611033.
Omeh, C. B., Olelewe, C. J., & Nwangwu, E. C. (2022). Impact of teaching computer programming using innovative pedagogy embedded with live online lectures and related tools: A randomized control trial. *Computer Applications in Engineering Education, 30*(5), 1390–1405.
Omeh, C. B., Olelewe, C. J., & Nwangwu, E. C. (2024). Fostering computer programming and digital skills development: An experimental approach. *Computer Applications in Engineering Education, 32*(2), Article e22711.
Omeh, C. B., Olelewe, C. J., & Ohanu, I. B. (2025). Impact of Artificial Intelligence Technology on Students' Computational and Reflective Thinking in a Computer Programming Course. *Computer Applications in Engineering Education*, *33*(3), e70052.
Parambil, A. J., Mishra, A. S., Chakravarty, S., & Pingle, Y. (2024). Saarthi: A programming language designed to introduce coding to high schoolers. In: *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 384–389). IEEE.
Park, S. H., & Ertmer, P. A. (2007). Impact of problem-based learning (PBL) on teachers' beliefs regarding technology use. *Journal of Research on Technology in Education, 40*(2), 247–267.

Peng, P., & Kievit, R. A. (2020). The development of academic achievement and cognitive abilities: A bidirectional perspective. *Child Development Perspectives, 14*(1), 15–20.
Polat, E., & Yilmaz, R. M. (2022). Unplugged versus plugged-in: Examining basic programming achievement and computational thinking of 6th-grade students. *Education and Information Technologies, 27*(7), 9145–9179.
Psotka, J. (2013). Educational games and virtual reality as disruptive technologies. *Journal of Educational Technology & Society, 16*(2), 69–80.
Rose, S., Habgood, M. J., & Jay, T. (2017). An exploration of the role of visual programming tools in the development of young children's computational thinking. *Electronic Journal of E-Learning, 15*(4), 297–309.
Santagata, R., König, J., Scheiner, T., Nguyen, H., Adleff, A. K., Yang, X., & Kaiser, G. (2021). Mathematics teacher learning to notice: A systematic review of studies of video-based programs. *Zdm–mathematics Education, 53*(1), 119–134.
Sevin, R., & DeCamp, W. (2016). From playing to programming: The effect of video game play on confidence with computers and an interest in computer science. *Sociological Research Online, 21*(3), 14–23.
Siswanto, D. H. (2025). The impact of a collaborative problem-based learning on performance in inverse matrix learning, critical thinking skills, and student anxiety. *Contemporary Education and Community Engagement (CECE), 2*(1), 1–11.
Sun, L., Guo, Z., & Zhou, D. (2022). Developing K-12 students' programming ability: A systematic literature review. *Education and Information Technologies, 27*(5), 7059–7097.
Suraworachet, W., Seon, J., & Cukurova, M. (2024). Predicting challenge moments from students' discourse: A comparison of GPT-4 to two traditional natural language processing approaches. In: *Proceedings of the 14th Learning Analytics and Knowledge Conference* (pp. 473–485).
Tabachnick, B. G., & Fidell, L. S. (2007). *Using Multivariate Statistics* (5th ed.). Allyn and Bacon.
Thorgeirsson, S., & Su, Z. (2021). Algot: An educational programming language with human-intuitive visual syntax. In: *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 1–5). IEEE.
Umakalu, C. P. U., Chioma, P., Omeh, C. B., & Omeh, C. B. (2025). Impact of teaching computer robotics programming using hybrid learning in public universities in Enugu state. *Nigeria. Vocational and Technical Education Journal, 5*(1), 1–8.
Van Aelst, S., & Willems, G. (2011). Robust and efficient one-way MANOVA tests. *Journal of the American Statistical Association, 106*(494), 706–718. https://doi.org/10.1198/JASA.2011.TM09748
Ventura, M., Ventura, J., Baker, C., Viklund, G., Roth, R., & Broughman, J. (2015). Development of a video game that teaches the fundamentals of computer programming. In: *SoutheastCon 2015* (pp. 1–5). IEEE.
Viberg, O., Cukurova, M., Feldman-Maggor, Y., Alexandron, G., Shirai, S., Kanemune, S., et al. (2024). What explains teachers' trust in AI in education across six countries?. *International Journal of Artificial Intelligence in Education*, 1–29.
White, G. L., & Sivitanides, M. P. (2002). A theory of the relationships between cognitive requirements of computer programming languages and programmers' cognitive characteristics. *Journal of Information Systems Education, 13*(1), 59–66.
Wood, D. F. (2003). *Problem Based Learning. Bmj, 326*(7384), 328–330.
Xu, E., Wang, W., & Wang, Q. (2023). The effectiveness of collaborative problem solving in promoting students' critical thinking: A meta-analysis based on empirical literature. *Humanities and Social Sciences Communications, 10*(1), 1–11.
Yağcı, M. (2018). Web-mediated problem-based learning and computer programming: Effects of study approach on academic achievement and attitude. *Journal of Educational Computing Research, 56*(2), 272–292.
Yang, W., Zhang, X., Chen, X., Lu, J., & Tian, F. (2024). Based case based learning and flipped classroom as a means to improve international students' active learning and critical thinking ability. *BMC Medical Education, 24*(1), 759.
Yilmaz, R., & Yilmaz, F. G. K. (2023a). Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning. *Computers in Human Behavior: Artificial Humans, 1*(2), Article 100005.
Yilmaz, R., & Yilmaz, F. G. K. (2023b). The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education: Artificial Intelligence, 4*, Article 100147.
Yusuf, A., Bello, S., Pervin, N., & Tukur, A. K. (2024). Implementing a proposed framework for enhancing critical thinking skills in synthesizing AI-generated texts. *Thinking Skills and Creativity, 53*, Article 101619.
Zhong, X., & Zhan, Z. (2024). An intelligent tutoring system for programming education based on informative tutoring feedback: system development, algorithm design, and empirical study. *Interactive Technology and Smart Education*, (ahead-of-print).

Publisher's Note