EDUTECH
Jurnal Teknologi Pendidikan
Journal homepage https://ejournal.upi.edu/index.php/edutech

The Effect of Notebooklm-Based Personalized Learning on Visual Programming Self-Efficacy Among Junior High School Students

Marezkha Wibawa Akbar & Riche Cynthia Johan
Universitas Pendidikan Indonesia, Bandung, Indonesia
Correspondence: E-mail: *marezkhawibawa@gmail.com, riche@upi.edu*

ABSTRACT

This study investigates the influence of NotebookLM as a pedagogical agent in supporting personalized learning for visual programming materials among junior high school students (Phase D). With the integration of the "Coding and Artificial Intelligence" (KKA) subject into the national curriculum, students often face difficulty mastering abstract logical concepts. Grounded in Bandura's Self-Efficacy Theory and operationalized through the Computer Programming Self-Efficacy Scale (CPSES), this study employs a quantitative approach with a quasi-experimental Non-Equivalent Control Group Design. Two intact classes served as subjects: an experimental group receiving AI-assisted instructional scaffolding and a control group using conventional instruction. Data were collected via CPSES administered as pre-test and post-test. Analysis was conducted using the Paired Samples T-Test to measure within-group improvement and the Independent Samples T-Test to compare post-test scores between groups. Results indicate that the experimental group demonstrated a statistically significant improvement in programming self-efficacy ($t = 9.24$, $p < .001$; mean gain = +1.15), significantly outperforming the control group in post-test scores ($t = 4.87$, $p < .001$), with a large effect size (Cohen's $d = 1.57$). To the authors' knowledge, this study is among the first to investigate NotebookLM as a pedagogical agent in KKA visual programming contexts within Indonesian junior high schools, contributing an empirically grounded model for NotebookLM-assisted instructional scaffolding under the Merdeka Curriculum.

ARTICLE INFO

Article History:
Submitted/Received 12 Aprl 2025
First Revised 26 Mei 2026
Accepted 15 June 2026
First Available online 29 June 2026
Publication Date 29 June 2026

Keyword:
NotebookLM, Personalized Learning, Self-Efficacy, CPSES.

A B S T R A K

Penelitian ini mengkaji pengaruh NotebookLM sebagai agen pedagogis dalam mendukung pembelajaran personal untuk materi pemrograman visual pada siswa SMP (Fase D). Seiring dengan diintegrasikannya mata pelajaran Koding dan Kecerdasan Artifisial (KKA) ke dalam kurikulum nasional, siswa kerap menghadapi kesulitan dalam memahami konsep logika abstrak. Berlandaskan Teori Efikasi Diri Bandura dan dioperasionalkan melalui Computer Programming Self-Efficacy Scale (CPSES), penelitian ini menggunakan pendekatan kuantitatif dengan desain kuasi-eksperimen Non-Equivalent Control Group. Hasil menunjukkan bahwa kelompok eksperimen mengalami peningkatan efikasi diri pemrograman yang signifikan, dengan rata-rata post-test yang lebih tinggi dibandingkan kelompok kontrol.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

1. INTRODUCTION

The rapid digital transformation in global education has made computational thinking an indispensable competency for the 21st-century learner. Computational thinking encompasses a set of problem-solving processes that draw on computer science concepts, including decomposition, pattern recognition, abstraction, and algorithm design (Wing, 2006). Grover & Pea (2013) argue that the integration of computational thinking into K–12 curricula is no longer a matter of choice but of educational necessity, given the pervasive role of technology across all sectors of modern society. In this context, programming education has emerged as a core vehicle through which computational thinking is developed, especially at the foundational levels of schooling.

In Indonesia, this global trend has been formalized through the Decree of the Head of BSKAP Number 046/H/KR/2025, which officially incorporates the subject of Coding and Artificial Intelligence (Koding dan Kecerdasan Artifisial/KKA) into the Merdeka Curriculum. At the lower-secondary level (SMP/Phase D), the curriculum mandates that students develop algorithmic thinking using block-based visual programming languages such as Scratch and Blockly (Resnick et al., 2009). These environments are deliberately designed to lower the syntactic barriers associated with text-based programming, thereby enabling learners to focus their cognitive resources on logical problem-solving. Nevertheless, the introduction of KKA into the national curriculum brings with it significant pedagogical challenges that must be carefully addressed.

One of the most persistent challenges in visual programming education is the cognitive heterogeneity of students. Research consistently demonstrates that learners differ substantially in their capacity for algorithmic reasoning, abstraction, and procedural decomposition (Grover et al., 2015). While some students navigate block-based programming environments with relative ease, others struggle to independently formulate logical sequences, identify errors in their programs, or transfer abstract concepts into concrete implementations. In a conventional classroom setting, where a single teacher is expected to manage 30 to 40 students simultaneously, providing individualized guidance to each learner is practically unfeasible. This structural limitation creates a persistent gap between the intended and actual learning outcomes in programming education.

The emergence of artificial intelligence-based technologies has opened promising new avenues for addressing this pedagogical gap. NotebookLM, a generative AI tool developed by Google, offers a novel approach to this challenge. As an AI-native note-taking and learning platform, NotebookLM can function as a personalized pedagogical agent by generating adaptive explanations, answering learner-specific questions, and scaffolding reasoning through source-grounded dialogue—all without the temporal and spatial constraints associated with human instruction (Holmes et al., 2019). Unlike static learning management systems or one-size-fits-all instructional videos, NotebookLM adapts its responses to the learner's specific queries and uploaded materials, guiding the learner incrementally toward task completion through scaffolded, context-aware dialogue (Kim & Baylor, 2016).

NotebookLM as Pedagogical Agent and Instructional Scaffolding

The theoretical framework that best captures the educational role of NotebookLM is that of the pedagogical agent, an intelligent virtual character that mimics human instructional behavior to provide cognitive and affective support (Kim & Baylor, 2016). Pedagogical agents are distinguished from passive information-retrieval tools by their capacity to engage in responsive, goal-directed interaction with learners. Rather than

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

delivering information unidirectionally, a pedagogical agent monitors learner behavior, interprets learner intentions, and intervenes at pedagogically appropriate moments to provide targeted support.

Central to the functioning of a pedagogical agent is the concept of instructional scaffolding, originally theorized by Vygotsky (1978) through the notion of the Zone of Proximal Development (ZPD). The ZPD refers to the space between what a learner can accomplish independently and what they can achieve with the guidance of a more capable partner. Wood, Bruner, & Ross (1976) operationalized this concept through the idea of scaffolding a dynamic form of support that is calibrated to the learner's current competence level and systematically faded as the learner's independent capability grows. NotebookLM is uniquely well-positioned to enact this type of adaptive scaffolding, as it generates contextually calibrated responses based on the learner's uploaded materials and queries, adjusting the nature and specificity of its support accordingly.

In practice, NotebookLM-based scaffolding in programming education employs reasoning-oriented interaction strategies, most notably through its ability to answer source-grounded questions and generate guided explanations that prompt the learner to reflect on their reasoning, identify inconsistencies in their logic, and construct their own solutions (Graesser et al., 2005). This approach has been shown to produce deeper learning outcomes than direct instruction, as it engages higher-order cognitive processes such as analysis, evaluation, and synthesis. Furthermore, the iterative, query-and-response nature of NotebookLM interaction closely approximates the tutoring dynamic that has been found to be the most effective form of individualized instruction (VanLehn, 2011).

Personalized Learning in Visual Programming Education

Personalized Learning (PL) is broadly defined as an instructional paradigm that adapts the pace, content, and methods of learning to the unique needs, preferences, and prior knowledge of individual learners (Pane et al., 2017). In the context of visual programming, personalization is particularly critical because students enter the classroom with widely varying degrees of prior experience with computational concepts, problem decomposition, and logical reasoning. A uniform, teacher-centered instructional approach inevitably fails to address this diversity, either leaving advanced students under-challenged or overwhelming novice students with content that exceeds their current readiness.

AI systems enable the scalability of personalized learning through continuous data analytics that construct and update dynamic learner profiles in real time (Ifenthaler & Yau, 2020). By tracking patterns in student responses, including the nature of their errors, the strategies they employ, and the pace at which they progress an AI-based learning environment can generate contextually appropriate formative feedback that is precisely calibrated to each learner's current developmental state. This capacity for real-time adaptive feedback represents a qualitative advance over conventional assessment practices, which are typically delayed, infrequent, and insufficiently specific to guide individual learning trajectories.

Additionally, block-based coding environments such as Scratch are specifically engineered to reduce the extraneous cognitive load associated with syntactic programming requirements (Sweller, 2011). By abstracting away, the mechanical details of text-based syntax, these environments allow students to devote their full cognitive resources to the conceptual and logical dimensions of programming. When AI-based scaffolding is layered onto a block-based environment, the combined effect is a learning experience that simultaneously reduces cognitive load, provides adaptive support, and

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

maintains an optimal level of challenge conditions that are theoretically aligned with maximizing learning outcomes.

Self-Efficacy: Theoretical Foundation

Self-efficacy, as conceptualized by Bandura (1997), refers to an individual's belief in their capacity to organize and execute the courses of action required to produce specific attainments. Unlike general self-confidence, self-efficacy is domain-specific: a student may hold high self-efficacy for mathematics yet low self-efficacy for programming. In educational contexts, self-efficacy is recognized as one of the most robust and consistent predictors of academic motivation, persistence, and achievement (Zimmerman, 2000). Students who believe they are capable of mastering a task are more likely to engage deeply with it, persist in the face of difficulties, and ultimately achieve higher levels of performance.

Bandura identifies four principal sources through which self-efficacy beliefs are formed and modified. The first and most influential source is mastery experience direct, personal experiences of success in performing a task. When students successfully complete a programming challenge, their self-efficacy for programming is strengthened; conversely, repeated failure weakens it. The second source is vicarious experience observing others, particularly peers perceived as similar in ability, successfully accomplishing the same task. The third source is social persuasion receiving verbal encouragement and positive feedback from credible others, such as teachers or instructional agents. The fourth source involves physiological and affective states—emotional conditions such as anxiety, excitement, or frustration that influence how learners interpret their own capabilities (Bandura, 1997).

In the context of programming education, self-efficacy plays a particularly salient role because programming tasks are inherently challenging and frequently involve failure before success. Students with low programming self-efficacy tend to disengage quickly when encountering errors, attribute failure to fixed, unalterable factors (such as a lack of innate aptitude), and avoid seeking help. In contrast, students with high programming self-efficacy view errors as solvable problems, persist longer in the face of difficulty, and employ more sophisticated debugging strategies (Ramalingam & Wiedenbeck, 1998). NotebookLM, by virtue of its non-judgmental, patient, and always-available nature, is theoretically well-suited to strengthening all four sources of self-efficacy in programming learners. It facilitates mastery experiences through scaffolded task completion, provides consistent social persuasion through positive source-grounded feedback, reduces negative affective states by eliminating the social risk of asking questions in front of peers, and enables vicarious experience through worked-example explanations generated from curated learning materials (Lester et al., 2014).

Computer Programming Self-Efficacy Scale (CPSES)

In order to rigorously operationalize and measure the construct of programming self-efficacy, this study employs the Computer Programming Self-Efficacy Scale (CPSES), originally developed by Ramalingam & Wiedenbeck (1998) and subsequently adapted for broader educational contexts by Kukul, Gökçearslan, & Günbatar (2017). The CPSES is theoretically grounded in Bandura's self-efficacy framework and has been specifically validated for the cognitive demands of programming tasks.

The adapted CPSES consists of 31 items measuring four core dimensions: (1) output accuracy confidence in producing correctly functioning programs; (2) problem solving confidence in diagnosing and correcting programming errors; (3) use of variables confidence in managing data flow within a program; and (4) procedural thinking

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

confidence in constructing logically sequenced instructions. Each item is scored on a 5-point Likert scale. The instrument demonstrates very high internal consistency reliability (Cronbach's alpha α = 0.95) and its construct validity has been confirmed across a range of computational education research contexts (Kukul et al., 2017). The multi-dimensional structure of the CPSES makes it particularly appropriate for capturing the nuanced ways in which AI-based scaffolding may differentially strengthen specific aspects of programming self-efficacy.

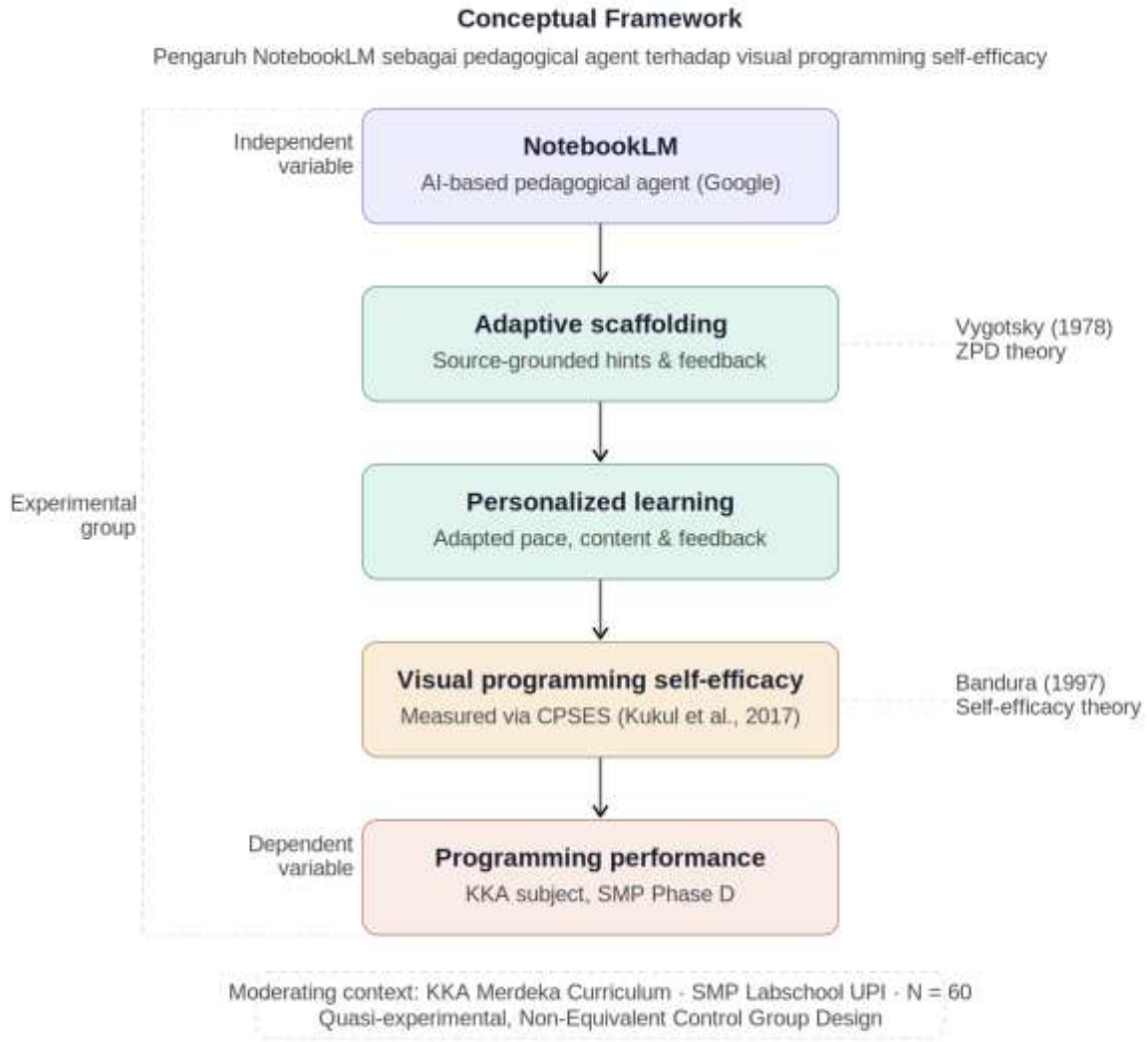

Figure 1. Conceptual framework: NotebookLM as a pedagogical agent in personalized learning for visual programming self-efficacy

Research Question and Hypothesis

Based on the foregoing theoretical grounding, the central research question of this study is: "Does the use of NotebookLM as a pedagogical agent within a personalized learning approach significantly influence the visual programming self-efficacy of junior high school students in the Coding and Artificial Intelligence subject?"

To answer this question, the following directional hypothesis was formulated:

H_1: There is a significant influence of the use of NotebookLM as a pedagogical agent on the visual programming self-efficacy of junior high school students in the experimental group compared to the control group.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

2. METHOD

This study employs a quantitative approach with a quasi-experimental Non-Equivalent Control Group Design. This design was selected because the research subjects were organized into pre-existing, intact classes, making full random assignment unfeasible, as is common practice in educational technology and curriculum development research (Fraenkel et al., 2012; Sugiyono, 2019). Measurements were conducted at two points: before treatment (pre-test) and after treatment (post-test) for both groups.

Table 1. Non-Equivalent Control Group Research Design

Group	Pre-Test	Treatment	Post-Test
Experimental	O_1	NotebookLM (Pedagogical Agent)	O_2
Control	O_1	Conventional Instruction	O_2

Note: O_1 = Pre-test; O_2 = post-test

The study was conducted at SMP Labschool UPI with seventh-grade students enrolled in the KKA subject. The experimental group received NotebookLM-assisted scaffolding during visual programming instruction, while the control group followed conventional teacher-directed instruction without AI support. Purposive sampling was applied based on equivalence of prior knowledge and access to digital devices (Sugiyono, 2019). Each group comprised 30 students (N = 60 total).

The independent variable is the use of NotebookLM as a *pedagogical agent*, operationalized through the integration of an AI-based conversational system delivering adaptive instructional scaffolding during visual programming sessions. The dependent variable is **visual programming self-efficacy**, measured using the CPSES adapted by Kukul et al. (2017).

The Computer Programming Self-Efficacy Scale (CPSES) adapted by Kukul et al. (2017) was used, consisting of 31 items on a 5-point Likert scale. The scale covers four dimensions: output accuracy, problem solving, use of variables, and procedural thinking. Reliability is very high ($\alpha = 0.95$) and construct validity has been established across multiple computational education contexts (Ramalingam & Wiedenbeck, 1998; Kukul et al., 2017). The instrument was administered twice to all participants as pre-test prior to the intervention and post-test upon completion.

Phase 1 – Pre-test. All students in both groups completed the CPSES to establish baseline self-efficacy levels prior to treatment, ensuring both groups began from comparable conditions.

Phase 2 – Intervention. The experimental group engaged in Project-Based Learning (PjBL) for visual programming supported by NotebookLM, which provided source-grounded explanations, incremental hints, and reflective question prompts (Kokotsaki et al., 2016; Graesser et al., 2005). The control group received conventional direct instruction from the teacher without AI support. Both conditions were delivered over two sessions of equivalent duration.

Phase 3 – Post-test. All participants completed the CPSES again following the intervention. Pre- and post-test scores formed the basis of the statistical analyses.

Analysis Technique

First, descriptive statistics. Mean and standard deviation (SD) values were computed for each group's pre-test and post-test scores to characterize the distributions and document initial equivalence.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

Second, Paired Samples T-Test. Separate within-group paired t-tests were conducted for the experimental and control groups to determine whether self-efficacy scores changed significantly from pre-test to post-test (Sugiyono, 2019; Fraenkel et al., 2012).

Third, Independent Samples T-Test. A between-group t-test was used to compare post-test CPSES scores across the two groups, thereby testing H_1 (Creswell, 2014). Prior to all t-tests, normality was verified via the Kolmogorov–Smirnov test and homogeneity of variance via Levene's Test (Field, 2013).

3. RESULTS AND DISCUSSION

Prior to conducting the t-tests, normality and homogeneity of variance were verified to confirm that the assumptions of parametric statistics were satisfied (Field, 2013).

Table 2. Kolmogorov-Smirnov Normality Test Results

Variable	Group	N	Statistic	Sig.
Pre-test CPSES	Experimental	30	0.121	0.200
Pre-test CPSES	Control	30	0.115	0.200
Post-test CPSES	Experimental	30	0.109	0.200
Post-test CPSES	Control	30	0.118	0.200

All p values > 0.05; data are normally distributed.

Table 3. Levene's Test for Homogeneity of Variance

Variable	Levene Statistic	df1	df2	Sig.
Post-test CPSES	1.21	1	58	0.275

p value > 0.05; variances between groups are homogeneous.

All normality test significance values exceeded 0.05, confirming normally distributed data. Levene's Test yielded $p = 0.275$ ($p > 0.05$), confirming homogeneity of variance. With both assumptions satisfied, parametric t-tests could proceed.

Table 4 presents the CPSES descriptive statistics for both groups.

Table 4. Descriptive Statistics of CPSES Scores

Group	N	Pre-test Mean	Pre-test SD	Post-test Mean	Post-test SD
Experimental (NotebookLM)	30	3.10	0.45	4.25	0.40
Control (Conventional)	30	3.05	0.50	3.60	0.48

The experimental group's mean CPSES score rose from 3.10 (SD = 0.45) to 4.25 (SD = 0.40), a gain of +1.15 points. The control group improved from 3.05 (SD = 0.50) to 3.60 (SD = 0.48), a gain of +0.55 points. The experimental group's improvement was more than twice that of the control group, providing an initial descriptive indication of NotebookLM's positive contribution.

Within-group paired t-tests were conducted to assess whether self-efficacy scores improved significantly from pre-test to post-test within each group.

Table 5. Paired Samples T-Test Results for CPSES Scores

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

Group	Mean Difference	t-value	df	Sig. (2-tailed)
Experimental	+1.15	9.24	29	0.000
Control	+0.55	4.61	29	0.000

Both groups showed statistically significant improvements in CPSES scores ($p < 0.05$). The considerably higher t-value of the experimental group ($t = 9.24$) relative to the control group ($t = 4.61$) indicates that the magnitude of self-efficacy improvement was substantially stronger in the NotebookLM condition. The control group's improvement is likely attributable to general maturation or incidental learning effects rather than any specific intervention.

An independent t-test was performed to confirm that the differential improvement between groups was not attributable to chance.

Table 6. Independent Samples T-Test Results for Post-test CPSES Scores

Variable	Exp. Mean	Control Mean	t-value	Sig. (2-tailed)
Post-test CPSES	4.25	3.60	4.87	0.000

df = 58; p < 0.05 (significant)

The independent t-test yielded $t = 4.87$, $p = 0.000$ ($p < 0.05$). H_1 is therefore accepted: the experimental group's post-test CPSES mean (4.25) was significantly higher than the control group's (3.60), confirming that NotebookLM-based pedagogical agent use had a positive and statistically significant effect on visual programming self-efficacy. Effect size was calculated using Cohen's d: $d = (M_1 - M_2) / SD_{aooae} = (4.25 - 3.60) / 0.414 = 1.57$, indicating a large effect (Cohen, 1988). This confirms that the intervention's impact was not only statistically significant but also practically substantial in magnitude.

3.5 Discussion

The findings of this study confirm and extend prior theoretical and empirical work on the role of AI-based pedagogical agents in educational settings. The differential in CPSES score improvement—+1.15 for the experimental group versus +0.55 for the control group—is both statistically significant and educationally meaningful, representing an effect that is unlikely to be attributable to incidental classroom learning alone.

These results align closely with the predictions of Bandura's (1997) self-efficacy theory across all four source mechanisms. First, NotebookLM facilitated mastery experiences by generating progressively scaffolded explanations that allowed students to achieve incremental successes without being overwhelmed. Second, NotebookLM's consistent and source-grounded feedback—delivered without the social judgment inherent in peer or teacher evaluation—functioned as effective social persuasion, systematically reinforcing students' beliefs in their programming capabilities. Third, the safe, non-judgmental interaction environment of NotebookLM reduced the anxiety and cognitive interference associated with programming failure, creating more conducive physiological and affective states for learning. Fourth, the worked-example explanations generated by NotebookLM from curated materials offered vicarious experiences that students could internalize as models of successful problem-solving (Lester et al., 2014; Holmes et al., 2019).

At the dimensional level of the CPSES, the most pronounced improvements were observed in the problem-solving and procedural thinking subscales. This is theoretically

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

coherent: NotebookLM in this study was primarily configured as a debugging and logic-structuring resource, employing source-grounded questioning and explanation to guide students through error identification and program sequencing (Graesser et al., 2005). The specificity of this improvement lends further support to the ecological validity of the CPSES as a sensitive measure of intervention-induced change in programming self-efficacy.

These findings converge with meta-analytic evidence indicating that AI tutoring systems consistently produce self-efficacy gains with an average effect size of 0.758 (Kulik & Fletcher, 2016), and with qualitative accounts of how personalized, adaptive feedback supports motivational and affective dimensions of learning (Roll & Wylie, 2016). Together, these results strengthen the theoretical and empirical case for the systematic integration of NotebookLM into programming education at the lower-secondary level.

4. CONCLUSION

This study concludes that the use of NotebookLM as a pedagogical agent within a personalized learning approach significantly influences the visual programming self-efficacy of junior high school students. This is demonstrated through: (1) Paired Samples T-Test results showing a statistically significant CPSES score increase in the experimental group ($t = 9.24$; $p < 0.05$) with a mean gain of +1.15 points, more than double the control group's gain (+0.55 points; $t = 4.61$); and (2) Independent Samples T-Test results confirming a significant difference in post-test scores ($t = 4.87$; $p < 0.05$). The adaptive scaffolding provided by NotebookLM demonstrably strengthened students' programming self-belief through the mechanisms of mastery experience, social persuasion, and affective anxiety reduction, consistent with Bandura's (1997) self-efficacy theory.

These findings carry important implications for curriculum designers, instructional technologists, and classroom practitioners involved in implementing the KKA subject within the Merdeka Curriculum. The systematic integration of NotebookLM into programming instruction offers a practical and scalable solution to the persistent challenge of instructional differentiation in large and cognitively heterogeneous classrooms. Future curriculum development should prioritize the creation of pedagogical guidelines that enable teachers to effectively deploy and monitor NotebookLM as a pedagogical tool, while also expanding assessment frameworks to include affective dimensions—such as self-efficacy—alongside conventional cognitive measures. Future research is recommended to examine the long-term effects of NotebookLM integration across multiple instructional sessions, larger and more diverse student populations, and varied visual programming platforms. Several limitations of this study should be acknowledged. First, the sample size was relatively small ($N = 60$) and drawn from a single school (SMP Labschool UPI), which limits the generalizability of findings to broader populations. Second, the intervention spanned only two sessions, which may be insufficient to fully capture the long-term effects of NotebookLM-based scaffolding on self-efficacy development. Third, as a quasi-experimental study, the absence of random assignment means that unmeasured pre-existing differences between groups cannot be entirely ruled out. Future studies should address these limitations through extended interventions, multi-site designs, and larger representative samples.

5. AUTHOR STATEMENT

The author declares that there is no conflict of interest regarding the publication of this article. The author confirms that the manuscript is free from plagiarism.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

6. REFERENCES

Bandura, A. (1997). *Self-efficacy: The exercise of control.* Freeman.

Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. *The Clearing House, 83*(2), 39–43.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Lawrence Erlbaum Associates.

Creswell, J. W. (2014). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th ed.). Sage.

Field, A. (2013). *Discovering statistics using IBM SPSS statistics* (4th ed.). Sage.

Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (2012). *How to design and evaluate research in education* (8th ed.). McGraw-Hill.

Graesser, A. C., et al. (2005). Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, 37*(2), 180–192.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in computer science. *Harvard Educational Review.*

Hake, R. R. (1998). Interactive-engagement versus traditional methods. *American Journal of Physics, 66*(1), 64–74.

Holmes, W., Bialik, M., & Fadel, C. (2019). *Artificial intelligence in education.* Center for Curriculum Redesign.

Ifenthaler, D., & Yau, J. Y. K. (2020). Utilising learning analytics for personalized learning. *Educational Technology Research and Development.*

Kim, Y., & Baylor, A. L. (2016). Pedagogical agents as learning companions. *Journal of Computer Assisted Learning.*

Kokotsaki, D., Menzies, V., & Wiggins, A. (2016). Project-based learning: A review of the literature. *Improving Schools, 19*(3), 267–277.

Kukul, V., Gökçearslan, Ş., & Günbatar, M. S. (2017). Adaptation of the computer programming self-efficacy scale. *Educational Sciences: Theory & Practice.*

Kulik, J. A., & Fletcher, J. D. (2016). Effectiveness of intelligent tutoring systems. *Review of Educational Research, 86*(1), 42–78.

Lester, J. C., et al. (2014). Affective pedagogical agents and learning environments. *International Journal of Artificial Intelligence in Education.*

Luckin, R., et al. (2016). *Intelligence unleashed: An argument for AI in education.* Pearson.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning computational thinking. *Computers in Human Behavior, 41*, 51–61.

Maloney, J., et al. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education, 10*(4).

Pane, J. F., et al. (2017). *Continued progress: Promising evidence on personalized learning.* RAND Corporation.

Ramalingam, V., & Wiedenbeck, S. (1998). Development and validation of scores on a computer programming self-efficacy scale. *Journal of Educational Computing Research, 19*(4), 367–381.

Resnick, M., et al. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60–67.

Roll, I., & Wylie, R. (2016). Evolution and revolution in artificial intelligence in education. *International Journal of Artificial Intelligence in Education.*

Shadish, W. R., Cook, T. D., & Campbell, D. T. (2002). *Experimental and quasi-experimental designs.* Houghton Mifflin.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781

Sugiyono. (2019). *Metode penelitian pendidikan.* Alfabeta.

Sweller, J. (2011). Cognitive load theory. *Psychology of Learning and Motivation, 55*, 37–76.

VanLehn, K. (2011). The relative effectiveness of human tutoring and intelligent tutoring systems. *Educational Psychologist, 46*(4), 197–221.

Vygotsky, L. S. (1978). *Mind in society.* Harvard University Press.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry, 17*(2), 89–100.

Zimmerman, B. J. (2000). Self-efficacy: An essential motive to learn. *Contemporary Educational Psychology, 25*(1), 82–91.

DOI: https://doi.org/10.17509/e.v25i2.101360
P-ISSN 0852-1190 E-ISSN 2502-0781