e-ISSN 2549-7472
Edumatic: Jurnal Pendidikan Informatika
Volume 10, Issue 2, **370-379.**
https://e-journal.hamzanwadi.ac.id/index.php/edumatic
DOI: 10.29408/edumatic.v10i2.35489

An Educational Game Based on Discovery Learning for Improving Computational Thinking in Programming Education

Zhafira Muthia Rabbani [1,*], Eki Nugraha [1], Latifahny Aridia Alfitri [1], Rosita Indrianti [2]

[1] Universitas Pendidikan Indonesia, Indonesia
[2] Poltekkes Kemenkes Jakarta 2, Indonesia
[*] Corresponding author: Zhafira Muthia Rabbani, Universitas Pendidikan Indonesia, Indonesia
✉ zhafiramuthia120904@upi.edu

Received: 13 June 2026 | Revised: 23 June 2026 | Accepted: 14 July 2026 | Published: 10 August 2026

Abstract
Programming education plays a crucial role in improving Computational Thinking (CT). However, students often experience difficulties in understanding abstract programming concepts, particularly branching control structures, highlighting the need for more effective learning approaches. This study developed a discovery learning-based educational game that integrates discovery learning syntax with computational thinking indicators into gameplay. The game was developed using the Research and Development (R&D) method with the ADDIE model and achieved a feasibility score of 93.8% based on expert validation. Its effectiveness was evaluated using a one-group pretest-posttest design involving 29 high school students, with data analyzed using the wilcoxon signed-rank test and N-Gain analysis. The results showed a significant improvement in students' computational thinking skills, with an N-Gain score of 0.46 (medium). Pattern recognition showed greater improvement than abstraction, suggesting that repeated encounters with similar branching problems improved students' pattern recognition skills. Students also responded positively to the game, with a score of 74.94% (good). These findings suggest that integrating discovery learning syntax with computational thinking indicators into gameplay can support computational thinking improvement in programming education.

Keywords: branching control structures; computational thinking; discovery learning; educational game; programming education

To cite this article: Rabbani, Z. M., Nugraha, E., Alfitri, L. A., & Indrianti, R. (2026). An Educational Game Based on Discovery Learning for Improving Computational Thinking in Programming Education. *Edumatic: Jurnal Pendidikan Informatika*, *10*(2), 370–379. https://doi.org/10.29408/edumatic.v10i2.35489

INTRODUCTION

Computational Thinking (CT) is increasingly recognized as a key competency for academic success and future workforce (Bers et al., 2022). Consequently, it has become an essential skill in Informatics education that enables students to analyze complex problems, recognize recurring patterns, and develop algorithm-based solutions to address challenges in the digital era (Palop et al., 2025; Wu et al., 2024). According to Mueller et al. (2017), CT consists of four components: decomposition, abstraction, pattern recognition, and algorithmic thinking. Decomposition refers to the ability to break down a problem into smaller and more manageable elements. Abstraction is the ability to extract relevant information while disregarding irrelevant details. Pattern recognition refers to the ability to recognize similarities between previously encountered problems and new problems. Meanwhile, algorithmic

thinking is the ability to develop logical and organized steps for solving a problem. In Indonesia, CT has been incorporated into the Informatics curriculum and is closely related to algorithms and programming (AP). CT focuses on problem analysis and solution strategies, whereas AP focuses on implementing these strategies through computer programs (Duckworth & Fraillon, 2025). Furthermore, learning programming can support the development of students' CT skills (Giannakoulas & Xinogalos, 2024).

Among AP topics, branching control structures are fundamental concepts that students need to learn. Understanding branching control structures requires students to distinguish relevant information from irrelevant information, organize actions based on that information, and recognize relationships among different cases, all of which closely associated with CT skills. However, students often struggle with programming because it is abstract and complex (Cheah, 2020). One of the programming concepts that students find difficult is branching control structures, largely due to misconceptions about control-flow statements (Sychev & Denisov, 2023). This condition may be influenced by the continued use of traditional teaching methods, which are generally teacher-centered, dominated by lectures and provide limited opportunities for students to participate (Patil et al., 2023). Consequently, students tend to spend more time correcting syntax errors than developing solutions to programming problems (Hisamuddin & Siregar, 2024). In contrast, the Merdeka Curriculum promotes a student-centered approach in which learning emphasizes active student participation.

One learning model that corresponds to these characteristics is discovery learning, which encourages students to develop conceptual understanding through exploration. Through activities such as identifying problems, collecting and processing information, and drawing conclusions, students are actively involved in constructing their own understanding throughout learning process (Nadira et al., 2026). Previous studies have shown that the implementation of discovery learning can improve students' problem-solving and computational thinking skills (He et al., 2025; Windari et al., 2025). However, these studies focused mainly on mathematics education, with limited evidence from programming education.

Along with advances in technology, learning media have also evolved, one example being educational games. Educational games can provide interactive learning experiences that help students visualize abstract concepts while increasing engagement and motivation (Byusa et al., 2022; Li et al., 2024). In addition, today's students are highly familiar with games, making educational games a potentially effective medium for learning programming concepts. Previous studies have also shown that educational games can improve students' computational thinking skills in programming courses (Giannakoulas & Xinogalos, 2024). However, their effectiveness also depends on how learning activities are integrated into gameplay.

Although previous studies have shown the effectiveness of educational games in supporting computational thinking development and the positive impact of discovery learning on students' computational thinking skills, few studies have integrated discovery learning syntax into educational games for programming education. Existing educational games generally focus on gameplay activities without explicitly integrating discovery learning processes. What distinguishes this study from previous work is the systematic integration of discovery learning syntax with computational thinking indicators into gameplay. Therefore, this study aims to develop a discovery learning-based educational game for learning branching control structures and examine its effectiveness in improving students' computational thinking skills and their responses to the developed media.

METHOD

This study employed a Research and Development (R&D) approach using the ADDIE model, which consists of five stages: Analysis, Design, Development, Implementation, and Evaluation. The ADDIE model was selected because it gives a systematic and flexible

framework for developing educational games (Spatioti et al., 2022). To obtain initial evidence regarding the effectiveness of developed product, a one-group pretest-posttest design was employed by comparing students' CT scores before and after using the educational game.

The population consisted of all Grade XI students (Phase F) taking the Informatics subject. Purposive sampling was used to select participants following the recommendation of the subject teacher, considering that the students had acquired basic knowledge of algorithms and programming but had not yet studied the topic in depth, particularly the subtopic of branching control structures. The sample initially consisted of 36 students from Class XI-5. However, only 29 students completed both the pretest and posttest. Therefore, the final analysis was conducted using data from 29 participants.

Data were collected using expert validation sheets, a multiple-choice computational thinking test, and a student response questionnaire. The educational game was validated by an expert lecturer from Computer Science Education Study Program using the Multimedia Mania 2004 – Judges' Rubric developed by North Carolina State University, which assesses both instructional and multimedia quality. The computational thinking test was developed based on the learning objectives and computational thinking indicators proposed by Mueller et al. (2017). Prior to implementation, the instrument was administered to students outside the research sample and analyzed for validity, reliability, difficulty level, and discrimination index. The results were used to determine items included in the pretest and posttest. Table 1 presents the operational indicators.

Table 1. Operational indicators of computational thinking skills

Computational Thinking Indicator	Operational Indicator
Decomposition	Breaking down a issue into smaller and more manageable elements.
Abstraction	Identifying relevant and disregarding irrelevant information.
	Simplifying a problem by eliminating unnecessary details.
	Drawing conclusions based on available information.
Pattern Recognition	Identifying patterns, similarities, or relationships between problems.
	Applying prior knowledge or experience to solve new problems.
Algorithmic Thinking	Developing logical and systematic steps for solving a problem.

Students' responses were measured using the MEEGA+ questionnaire developed by Petri et al. (2024), which consists of 14 statements rated on a five-point Likert scale and is designed to evaluate educational games by measuring multiple dimensions of the player experience. The development process followed the ADDIE model. During the analysis stage, a literature review and field study were conducted to identify learning needs and challenges. The design stage involved preparing instructional materials and planning the educational game. During the development stage, educational game was created and validated by an expert lecturer. The implementation stage involved learning activities using the developed educational game, followed by the administration of pretests, posttests, and response questionnaires. Finally, the evaluation stage was conducted to assess the effectiveness of the educational game.

The collected data was analyzed quantitatively. The Shapiro-Wilk normality test was conducted at a significance level (alpha) of 0.05 because this test is suitable for samples with fewer than 50 participants. Since the data was not normally distributed, the difference between pre-test and post-test scores was analyzed using the Wilcoxon Signed-Rank Test. Students' CT skills improvement was measured using the normalized gain (N-Gain) proposed by Hake (1998), which is shown in equation 1.

$$< g > = \frac{(posttest\ score) - (pretest\ score)}{(maximum\ possible\ score) - (pretest\ score)} \quad (1)$$

The N-Gain values were categorized as high ($0.70 \leq g \leq 1.00$), medium ($0.30 \leq g \leq 0.69$), and low ($0.00 \leq g \leq 0.29$). Students' responses were analyzed by comparing the obtained scores with the ideal scores and converting them into percentages, which were categorized as poor (0%–24%), fair (25%–49%), good (50%–74%), and very good (75%–100%).

RESULTS AND DISCUSSION

Results

A literature review, teacher interviews, and student questionnaires were conducted to identify learning needs. The findings indicated that algorithms and programming were among the most challenging topics and that students expressed a strong interest in game-based learning media. To address these needs, a discovery learning-based educational game was designed for branching control structure learning by integrating discovery learning syntax with CT indicators into the gameplay. This integration enables each gameplay activity to support specific CT skills, as summarized in Table 2.

Table 2. Integration of discovery learning syntax and ct indicators in the educational game

Discovery Learning Syntax	Player Activities in the Game	CT Indicator
Stimulation and Problem Identification	Players receive stimuli through interactions with NPCs that present problems related to everyday life and the learning material. Players are required to identify the main problem and determine an initial solution.	Abstraction
Data Collection	Players explore the game environment and interact with NPCs to get learning materials progressively. Previously learned materials can also be accessed through the module as an additional learning resource.	Decomposition
Data Processing	Players complete challenges and quizzes related to the learning material. In the challenges, players encounter several problems that share similar patterns but involve different branching conditions.	Pattern Recognition, Decomposition, and Algorithmic Thinking
Verification	Players may use hints (clues) when they experience difficulties in completing the challenges.	Algorithmic Thinking
Conclusion	Players summarize the material they have learned and provide examples of everyday situations that can be solved using branching structures through the student worksheet (LKPD).	Abstraction and Pattern Recognition

Figure 1 presents the navigation structure of the educational game. The game implements a progressive unlock system in which learning content is accessed gradually according to students' learning progress. Each learning topic consists of three stages: Adventure, Material, and Quiz. Through these stages, students are guided to explore problems, learn branching control structure concepts, and apply their understanding through challenges and quizzes.

Completion of each stage contributes to students' progress and unlocks subsequent learning content.

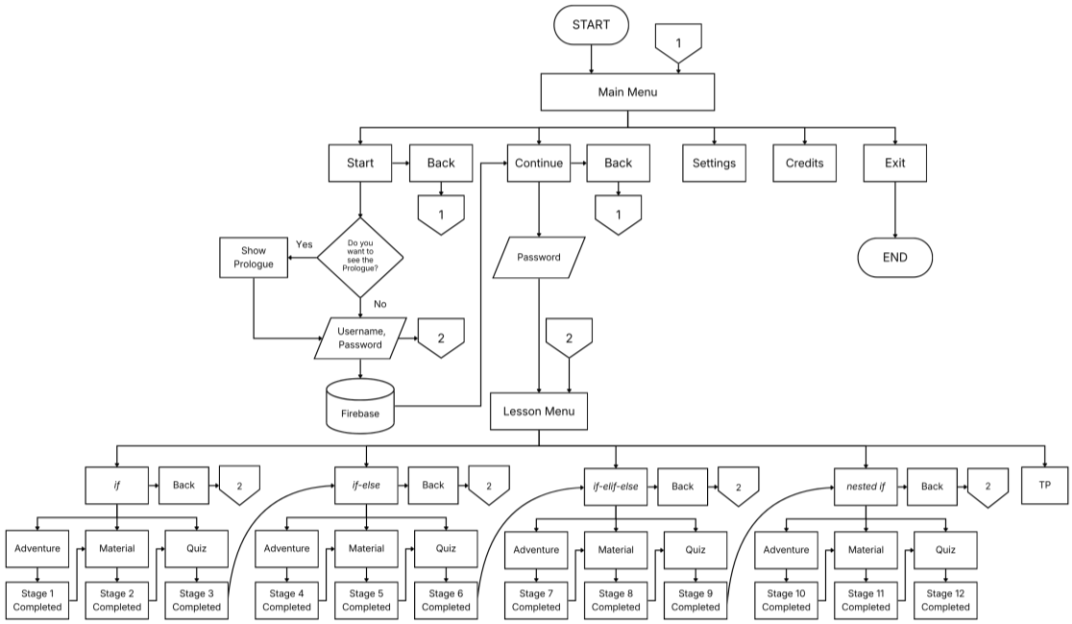


Figure 1. Navigation structure of the educational game

The developed educational game was subsequently validated by an expert lecturer from Computer Science Education Study Program using the Multimedia Mania 2004 – Judges' Rubric. The validation resulted in an overall feasibility score of 93.8% (very good), as presented in Table 3. Documentation received the lowest score (91.3%), mainly because the sources of the learning materials were not yet provided. The validator also suggested simplifying several navigation buttons. Accordingly, a Credits section was added, and the navigation buttons were simplified before implementation.

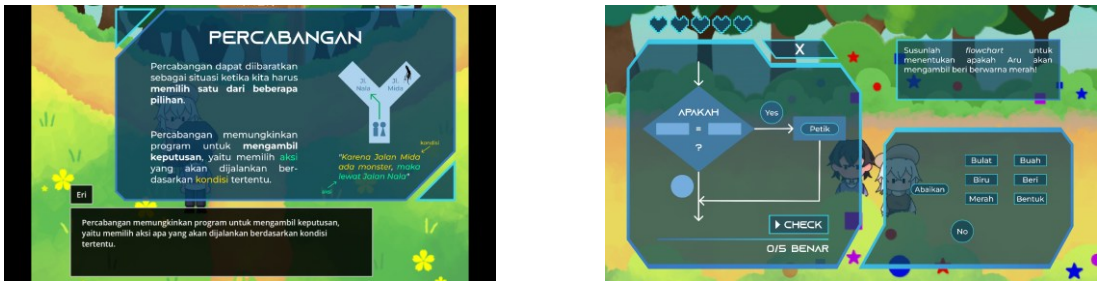
Table 3. Expert validation results of the educational game

Aspect	Percentage
Mechanics	93.1%
Multimedia Elements	95.0%
Information Structure	95.0%
Documentation	91.3%
Content Quality	93.5%
Overall Average	93.8%
Category	Very Good

Figure 2 presents examples of the developed educational game interface. As shown in Figure 2(a), learning materials are delivered through NPC-guided interactions. After studying the learning materials, students complete challenge activities by constructing flowcharts or Python code, as shown in Figure 2(b). These challenge activities require students to apply the concepts learned to solve branching problems.

Following expert validation and revision, the educational game was implemented with Grade XI students at SMA Negeri 5 Jakarta over four meetings. The implementation included a pretest, learning activities using the educational game supported by student worksheets (LKPD), a posttest, and the administration of the MEEGA+ questionnaire to collect students' responses. The effectiveness of the educational game was evaluated using pretest and posttest

data from the 29 students who completed the implementation phase. The data were analyzed using normality, hypothesis, and N-Gain analyses. The findings of the normality test performed using the Shapiro Wilk are summarized in Table 4.



(a) (b)

Figure 2. (a) Learning material interface; (b) Challenge interface

Table 4. Normality test results using the Shapiro Wilk Test

Test Type	n	Mean	W Statistic	p-value	Conclusion
Pretest	29	50.69	0.913	0.021	Not normally distributed
Posttest	29	75.69	0.936	0.077	Normally distributed

As presented in Table 4, the average pretest score was 50.69, while the average posttest score increased to 75.69. The Shapiro Wilk test showed that the pretest data were not normally distributed ($p = 0.021 < 0.05$), whereas the posttest data were normally distributed ($p = 0.077 > 0.05$). Therefore, the Wilcoxon-Signed Rank Test was used as a non-parametric alternative for further analysis. The results are summarized in Table 5.

Table 5. Hypothesis test results using the wilcoxon signed-ranks test

n	Pretest Mean	Posttest Mean	Z	Asymp. Sig. (2-tailed)
29	50,69	75,69	-4.380	<.001

As presented in Table 5, the Asymp. Sig. (2-tailed) value was less than 0.001, which is below the 0.05 significance level. This result indicates statistically significant difference between students' pretest and posttest scores after learning with the educational game. The calculated effect size ($r = 0.81$) indicates a large effect, suggesting that the educational game had a substantial impact on students' CT skills. To determine the improvement in students' CT skills, an N-Gain analysis was performed, and the results are summarized in Table 6.

Table 6. Students' n-gain results

N-Gain Category	Number of Students	Average N-Gain
High	8	0.83
Medium	13	0.49
Low	8	0.06
Overall	29	0.46 (Medium)

As presented in Table 6, all students showed improvements in their CT skills, with 8 students achieving high improvement, 13 achieving medium improvement, and 8 achieving low improvement. The overall N-Gain score was 0.46 (medium), indicating a moderate improvement in students' CT skills. The variation in learning gains may result from differences in students' prior programming knowledge and learning readiness. Further analysis of each CT indicator is presented in Table 7.

As shown in Table 7, all computational thinking indicators improved and were categorized within the medium category. Pattern recognition achieved the highest N-Gain score (0.45), whereas abstraction obtained the lowest (0.34), indicating that abstraction remained the most challenging CT indicator for students. Students' responses to the educational game were also analyzed using the MEEGA+ questionnaire, as presented in Table 8.

Table 7. N-Gain results for each computational thinking indicator

Computational Thinking Indicator	N-Gain	Category
Decomposition	0.35	Medium
Abstraction	0.34	Medium
Pattern Recognition	0.45	Medium
Algorithmic Thinking	0.44	Medium

Table 8. Students' responses to the educational game

Dimension	Percentage	Category
Usability	75.45%	Very Good
Player Experience	74.73%	Good
Overall Average	74.94%	Good

Discussion

The findings showed that discovery learning-based educational game achieved a very good level of feasibility, effectively improved CT skills in learning branching control structures, and received positive responses from students. The significant improvement from the pretest to the posttest and the overall N-Gain score of 0.46 (medium) indicate that integrating discovery learning syntax into gameplay provides a meaningful learning environment that helps the development of CT skills through active exploration and problem solving. Students' positive responses further suggest that the game provided a usable and engaging learning experience that supported the learning process. Overall, these findings support the research objective of developing a discovery learning-based educational game to improve CT skills and demonstrate the value of integrating discovery learning syntax into game design.

The improvement in students' CT skills can be explained by the learning mechanisms embedded within the educational game. Discovery learning encourages students to actively construct knowledge through exploration rather than passively receiving information, which is consistent with Bruner's theory of learning by doing (Ozdem-Yilmaz & Bilican, 2020). In this study, the discovery learning syntax was integrated into gameplay through stimulation and problem identification, exploration, challenges, immediate feedback, and conclusion activities. These activities guided students progressively throughout the learning process, helping them understand the concepts while independently solving branching problems. Immediate feedback and hints served as scaffolding, enabling students to evaluate and revise their solutions through trial and error while strengthening their understanding of the concepts being learned. Consequently, students actively engaged in exploration and problem solving throughout gameplay, allowing them to develop CT skills as part of the learning process. This finding is in line with Hanafi and Ratnasari (2026), who argued that learning improvement in educational games is influenced not only by instructional content but also by the learning mechanisms embedded within the game design.

During exploration, students repeatedly encountered branching problems with similar structures but different conditions. They first learned the basic concepts and flowcharts of branching control structures before progressing to flowchart challenges, branching syntax, and coding challenges, creating a staged learning sequence. Furthermore, solving these challenges

required students to break each problem into smaller elements, such as the required conditions, operators, and actions, before constructing flowcharts and Python code as solutions. After solving several similar problems, students recognized recurring solution patterns, making it easier to develop algorithms for subsequent problems. These findings suggest that decomposition, pattern recognition, and algorithmic thinking worked together throughout the problem-solving process. Among all indicators, abstraction showed the lowest improvement. This may be because students had to distinguish relevant information from contextual game narratives while simultaneously understanding learning concepts. According to cognitive load theory, presenting unnecessary information may increase extraneous cognitive load, thereby reducing learning performance (Darejeh et al., 2021). A comparable trend was reported by Monemnasi et al. (2026), who found that abstraction tends to improve less than other CT indicators in game-based learning environments.

Although the educational game significantly improved CT skills, the overall N-Gain remained within the medium category. This result may be explained by several factors, including the relatively short four-session intervention, differences in students' prior programming knowledge and learning readiness, and the need for more adaptive scaffolding to accommodate differences in students' learning pace during gameplay. Nevertheless, the findings are generally consistent with previous studies reporting the positive effects of educational games and discovery learning on CT development. Similar to Barrón-Estrada et al. (2022) and Crnković et al. (2024), this study found improvements in students' pattern recognition and algorithmic thinking skills through educational game activities. Likewise, Smetsers-Weeda and Smetsers (2017) reported that constructing flowcharts and program code helps students decompose problems and evaluate the algorithms they develop. However, unlike previous studies that examined educational games or discovery learning separately, this study integrated discovery learning syntax directly into gameplay while aligning each learning stage with CT indicators. This integration may explain why improvements were observed across multiple CT components rather than only in programming skills.

Students also responded positively to the educational game, indicating that it was well accepted as a learning medium. The very good level of usability enabled students to interact with the game with minimal difficulty, allowing them to focus on learning activities rather than game mechanics. Furthermore, the good level of player experience suggests that the educational game-maintained students' interest and active participation throughout the learning process. Students also perceived that the game had a clear connection to the learning material and helped them better understand the concepts being studied.

These findings suggest that educational game effectiveness relies not only on engaging gameplay but also on the integration of instructional activities that support learning objectives. This study contributes to programming education by demonstrating how discovery learning syntax can be integrated with CT indicators throughout gameplay. The proposed design further shows that CT development depends not only on gameplay itself but also on the integration of structured learning activities.

The findings should be interpreted in light of several methodological limitations. The one-group pretest–posttest design without a control group limits the extent to which improvements in CT skills can be attributed to the educational game. In addition, the four-session intervention involved students from a single class, limiting the generalizability of the findings. Future research should use experimental or quasi-experimental designs with larger, more diverse samples and longer interventions. Further studies could also examine different programming topics, compare discovery learning with alternative instructional models, and explore AI-based adaptive scaffolding for diverse learning needs.

CONCLUSION

This study demonstrated that integrating discovery learning syntax into an educational game effectively supported computational thinking development in learning branching control structures. The findings indicate that combining structured learning activities with gameplay creates a meaningful learning environment that encourages active exploration, problem solving, and knowledge construction. This study contributes to programming education by expanding the application of discovery learning in game-based learning and demonstrating how discovery learning syntax can be integrated with CT indicators throughout gameplay. The proposed design may serve as reference for Informatics teachers and educational game developers in designing educational games that support CT. Future research is encouraged to evaluate the proposed design across different programming topics, compare its effectiveness with other instructional models, and investigate the integration of AI-based adaptive scaffolding to better accommodate students' diverse learning needs.

REFERENCES

Barrón-Estrada, M. L., Zatarain-Cabada, R., Romero-Polo, J. A., & Monroy, J. N. (2022). Patrony: A mobile application for pattern recognition learning. *Educ Inf Technol*, *27*, 1237–1260. https://doi.org/10.1007/s10639-021-10636-7

Bers, M. U., Strawhacker, A., & Sullivan, A. (2022). *The state of the field of computational thinking in early childhood education* (OECD Education Working Papers No. 274). OECD Publishing. https://doi.org/10.1787/3354387a-en

Byusa, E., Kampire, E., & Mwesigye, A. R. (2022). Game-based learning approach on students' motivation and understanding of chemistry concepts: A systematic review of literature. *Heliyon*, *8*(5). https://doi.org/10.1016/j.heliyon.2022.e09541

Cheah, C. S. (2020). Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*, *12*(2), ep272. https://doi.org/10.30935/cedtech/8247

Crnković, V. M., Crnković, B., Traunkar, I., & Dlab, M. H. (2024). [AI] Explore!: An educational game for developing programming skills and algorithmic thinking. *Proceedings of the European Conference on Games Based Learning, 18*(1), 623–633. https://doi.org/10.34190/ecgbl.18.1.2866

Darejeh, A., Marcus, N., & Sweller, J. (2021). The effect of narrative-based E-learning systems on novice users' cognitive load while learning software applications. *Educational Technology Research and Development*, *69*, 2451–2473. https://doi.org/10.1007/s11423-021-10024-5

Duckworth, D., & Fraillon, J. (2025). Computational thinking framework. In J. Fraillon & M. Rožman (Eds.), *IEA International Computer and Information Literacy Study 2023* (pp. 39–56). Springer. https://doi.org/10.1007/978-3-031-61194-0_3

Giannakoulas, A., & Xinogalos, S. (2024). Studying the effects of educational games on cultivating computational thinking skills to primary school students: a systematic literature review. *Journal of Computers in Education*, *11*, 1283–1325. https://doi.org/10.1007/s40692-023-00300-z

Hake, R. R. (1998). Interactive-engagement versus traditional methods: A six-thousand student survey of mechanics test data for introductory physics courses. *American Journal of Physics*, *66*(1), 64–74. https://doi.org/10.1119/1.18809

Hanafi, S. C., & Ratnasari, C. I. (2026). Gamified Mobile Learning for Teaching the Prophet's Mandatory Traits in Elementary Education. *Edumatic: Jurnal Pendidikan Informatika*, *10*(1), 230–239. https://doi.org/10.29408/edumatic.v10i1.34206

He, X., Norulhuda, I., & He, M. (2025). The Effect of Problem-Based Learning, Cooperative Learning, and Discovery Learning on Mathematical Problem-Solving Skill: A Meta-

Analysis and Systematic Review. *Pertanika Journal of Social Science and Humanities*, *33*(3), 1037–1058. https://doi.org/10.47836/pjssh.33.3.05

Hisamuddin, M. Z., & Siregar, M. U. (2024). Evaluasi Penggunaan Flowgorithm dalam Pembelajaran Algoritma Pemrograman menggunakan TAM. *Edumatic: Jurnal Pendidikan Informatika*, *8*(1), 84–92. https://doi.org/10.29408/edumatic.v8i1.25413

Li, Y., Chen, D., & Deng, X. (2024). The impact of digital educational games on student's motivation for learning: The mediating effect of learning engagement and the moderating effect of the digital environment. *PLoS ONE*, *19*(1). https://doi.org/10.1371/journal.pone.0294350

Monemnasi, Y., Agustini, K., & Suartama, I. K. (2026). Efektivitas Multimedia Gamifikasi dalam Pembelajaran Berpikir Komputasional di SMP: Systematic Literature Review. *Cetta: Jurnal Ilmu Pendidikan*, *9*(2), 33–49. https://doi.org/10.37329/cetta.v9i2.5296

Mueller, J., Beckett, D., Hennessey, E., & Shodiev, H. (2017). Assessing computational thinking across the curriculum. In P. Rich & C. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 251–276). Springer. https://doi.org/10.1007/978-3-319-52691-1_16

Nadira, S., Erna, M., & Herdini, H. (2026). Implementation of The Discovery Learning Model to Improve The Activity and Learning Outcomes. *Edunesia: Jurnal Ilmiah Pendidikan*, *7*(1), 413–426. https://doi.org/10.51276/edu.v7i1.1415

Ozdem-Yilmaz, Y., & Bilican, K. (2020). Discovery learning—Jerome Bruner. In B. Akpan & T. J. Kennedy (Eds.), *Science education in theory and practice* (pp. 177–190). Springer. https://doi.org/10.1007/978-3-030-43620-9_13

Palop, B., Díaz, I., Rodríguez-Muñiz, L. J., & Santaengracia, J. J. (2025). Redefining computational thinking: A holistic framework and its implications for K-12 education. *Educ Inf Technol*, *30*(10), 13385–13410. https://doi.org/10.1007/s10639-024-13297-4

Patil, P., Jadhav, P., & Kanase, M. (2025). Imparting Effective Teaching Learning Methods for Teaching C Programming Course to First Year Non-IT Students. *Journal of Engineering Education Transformations*, *36*, 133–140. https://doi.org/10.16920/jeet/2023/v36is2/23019

Petri, G., Wangenheim, C. G., & Borgatto, A. F. (2024). MEEGA+: Systematic model to evaluate educational games. In N. Lee (Ed.), *Encyclopedia of computer graphics and games* (pp. 1112–1119). Springer. https://doi.org/10.1007/978-3-031-23161-2_214

Smetsers-Weeda, R., & Smetsers, S. (2017, October). Problem solving and algorithmic development with flowcharts. *In Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 25–34). Association for Computing Machinery. https://doi.org/10.1145/3137065.3137080

Spatioti, A. G., Kazanidis, I., & Pange, J. (2022). A comparative study of the ADDIE instructional design model in distance education. *Information*, *13*(9), 402. https://doi.org/10.3390/info13090402

Sychev, O., & Denisov, M. (2023). Explain trace: Misconceptions of control-flow statements. *Computers*, *12*(10), 192. https://doi.org/10.3390/computers12100192

Windari, I., Fatra, M., & Diwidian, F. (2025). Discovery Learning: Its Impact on Students' Mathematical Computational Thinking Ability and Self-Confidence. *JNPM (Jurnal Nasional Pendidikan Matematika)*, *9*(1), 51–65. https://doi.org/10.33603/jnpm.v9i1.10423

Wu, T. T., Asmara, A., Huang, Y. M., & Permata, H. I. (2024). Identification of Problem-Solving Techniques in Computational Thinking Studies: Systematic Literature Review. *Sage Open*, *14*(2). https://doi.org/10.1177/21582440241249897