ISSN: 2289-7844 | e-ISSN: 0127-9750
DOI: https://doi.org/10.37134/jictie.vol12.2.6.2025

Journal of ICT in Education (JICTIE)
Volume 12, Issue 2, Pages 88-98, 2025

Research Article

The Development of Educational Programming Software (CTCode) to Enhance Computational Thinking Skills for Novice Students

Nurul Najwa Nabihah Sa'adon[1], Nor Hasbiah Ubaidullah[1]*, Teddie Darmizal[2]

[1]*Faculty of Computing and Meta-Technology, Universiti Pendidikan Sultan Idris, Perak, Malaysia; wawanurnabihah@gmail.com, hasbiah@meta.upsi.edu.my*
[2]*Department of Informatics Engineering-Faculty of Science and Technology, Universitas Islam Negeri Sultan Syarif Kasim Riau, Indonesia; teddie.2405@gmail.com*

Received: 30 August 2024; Revised: 27 August 2025; Accepted: 23 September 2025; Published: 2 October 2025

**corresponding author*

Abstract

This study aimed to develop a programming learning software designed to enhance computational thinking skills among novice students and to evaluate the effectiveness of the CTCode software. The development of CTCode was guided by the Scrum and ASSURE models. The evaluation utilized an experimental method involving two groups of students, a treatment group and a control group, with 30 participants in each group. An assessment instrument consisting of a pre-test and post-test was used. The test consisted of 13 questions divided into three sections: Part A (objective questions), Part B (subjective questions), and Part C (code output). The results, analyzed using a simple t-test, yielded a t-value of 3.531. This indicates a significant difference in scores between students who used the newly developed software, based on computational thinking teaching and learning models, and those who used existing software. In conclusion, the study demonstrates that students enrolled in programming courses benefit from using CTCode. The software's activities, such as quiz, drag and drop, and mix and match, incorporate metacognitive techniques and problem-solving strategies, which can effectively enhance students' computational thinking skills. The findings suggest that CTCode can serve as a valuable resource for fostering an engaging and structured learning environment in programming education.

Keywords: computational thinking skills, metacognitive, problem-solving, learning software.

INTRODUCTION

Computational thinking (CT) is a crucial skill that students should master in the 21st century to effectively tackle and resolve complex problems in various fields (Nouri et al., 2020). Grover et al. (2023) state that integrating computational thinking into the curriculum enhances the understanding of computer science concepts. In the field of Computer Science, a subject closely related to computational thinking is programming. According to Wan Chong Chai (2022), programming emphasizes computational thinking, which encompasses the ability to analyze, design, create algorithms, and solve problems. Learning programming can be challenging and often demands strong metacognitive and problem-solving skills. Metacognitive skills and problem-solving skills are interconnected through processes like reasoning, understanding, and information gathering (Mohd Jalaluddin & Said, 2022). The application of metacognition in programming demonstrates significant advancements in establishing a robust conceptual

and strategic knowledge foundation for program design (Juhaina et al., 2020), while students who possess strong problem-solving skills are valuable for logically analyzing programming challenges. The use of educational software that incorporates game elements has been around for some time, combining play with learning (Zolkipli et al., 2023). This approach fosters a fun learning environment while enhancing collaborative skills among students (Norlizawaty et al., 2021). The use of games in educational software encourages students to focus and think creatively while playing. Complex thinking is a vital component of 21st-century learning, where students who excel in higher-order thinking skills can create a society that is both competitive and courageous in confronting challenges.

LITERATURE REVIEW

Programming

Programming is regarded as a crucial skill in today's society. However, it is commonly understood that learning programming can be challenging, especially for novice students with diverse backgrounds and experiences (Maria et al., 2022). According to Piumi et al. (2021), novice students face significant challenges in learning programming due to the variety of programming languages they need to master in order to establish a more conducive learning environment. According to Johan (2021), teaching and learning programming present one of the greatest challenges in computer education. Therefore, in this study, the researcher has selected a programming course to investigate ways to help novice students enhance their computational thinking skills through the use of developed software.

Computational Thinking

Computational thinking skills are essential in the fields of computer science and information technology. Recent studies have highlighted the importance of developing these skills among university and college students, particularly through programming courses (Youngjun et al., 2020; Nor Hasbiah et al., 2021). There is a strong connection between computational thinking (CT) and programming. Engaging in programming significantly enhances CT, which involves logical reasoning, abstract thinking, and effective problem-solving skills (Chan et al., 2023).

In this study, the researcher developed a software program to enhance computational thinking skills among novice students. By engaging with the activities within the software, students can indirectly strengthen their computational thinking abilities. This occurs because completing the tasks associated with each activity incorporates metacognitive and problem-solving techniques, both of which contribute to the improvement of computational thinking skills. Computational thinking skills are essential for new programmers to effectively solve problems related to coding computer systems (Nurbaya & Mohd Effendi, 2023). Programming and computational thinking activities are interconnected, as both are crucial for creating effective programs. Consequently, coding provides teachers with a means to instruct and empowers students to acquire computational thinking skills.

Metacognitive

Metacognition refers to the awareness and understanding of one's own cognitive processes. It has been linked to academic performance at all educational levels, including higher education institutions (Jayne & Matt, 2022). Metacognition consists of two components: metacognitive knowledge, which pertains to how students understand their learning processes, and metacognitive strategies, which involve planning, monitoring, and evaluating their learning (Mohammed, 2023). According to Unal and Er (2020), metacognitive skills are essential for learning activities like programming. Programmers must organize

their approach, select appropriate strategies, and structure their programming knowledge throughout the process. Developing these metacognitive skills is crucial for effectively solving programming problems. In a study conducted by Nor Hasbiah et al. (2020) on the relationship between metacognitive techniques and problem-solving, it was found that these metacognitive skills can be integrated with elements of computational thinking skills to enhance the computational thinking abilities of novice students.

This study incorporates metacognitive techniques into the learning software. Metacognition aids students in planning their actions, developing strategies, and evaluating their performance during activities. In this program, students will utilize these techniques to tackle programming problems presented at each level of the activities. Employing metacognitive strategies to solve the challenges in each activity enhances students' computational thinking skills. Learning computer programming courses can be challenging, especially for beginners. Students with strong metacognitive skills can recognize concepts that others find challenging and select effective strategies to grasp those concepts (Julie et al., 2021). Metacognition is crucial in learning programming, as metacognitive techniques aid in solving programming problems effectively.

Problem-Solving

Riyadi et al. (2021) state that problem-solving is a critical skill required in 21st-century education, as it is essential for addressing various challenges that arise. A problem-centered learning approach enhances the effectiveness of learning (Ssemugenyi, 2022). Programming allows individuals to create new digital solutions, address challenges, and bring ideas to life (Jalal et al., 2020). However, a lack of effective problem-solving skills in this field can hinder one's ability to make sound decisions (Siti Nordianah et al., 2020). According to Akben (2020), it is essential for teachers to be aware of the appropriate problem-solving approaches that cater to their students' needs.

In this study, the learning software developed includes various activities aimed at teaching programming. Problem-solving is crucial for tackling programming challenges, such as algorithms and root cause analysis (Sudin et al., 2022). To assist students in completing the assigned tasks within the CTCode learning software, problem-solving techniques are integrated into all activities. Problem-solving skills are an essential aspect of programming education, involving the process of developing solutions to programming challenges (Sohail et al., 2021). To effectively solve these problems, students must engage in several key activities, including analyzing the problem, planning a solution, designing the approach, coding the solution, and evaluating the results (Zamzuri et al., 2024; Azman et al., 2024). Each of these activities is a critical component of the problem-solving techniques that students need to apply before executing their work.

Gamification in Learning

Educational game applications have rapidly become essential pedagogical tools that facilitate and motivate learning (Samantha & Ronel, 2021). In education, games positively impact learning beyond just making it enjoyable (Athanasios & Stylianos, 2023). Incorporating game elements makes subjects more engaging and easier for students to understand. This approach captures students' interest and encourages their active involvement in the teaching and learning process (Nurul Dayana & Maizatul et al, 2020). Furthermore, using games in learning enhances problem-solving skills. Ignacio (2021) states that game-based learning is a platform that fosters imagination and divergent thinking skills necessary for solving problems and communicating ideas.

The CTCode learning software has been developed to include a variety of activities that incorporate game elements. These games are designed to support metacognitive techniques while enhancing computational thinking skills. According to Agbo et al. (2021), game-based learning activities, when combined with computational thinking skills, not only boost student interest but also improve computational thinking abilities, laying a solid foundation for solving programming problems.

METHODOLOGY

The methodology for this study is divided into two: software development and evaluation. Therefore, this study is focused on the development of a CTCode learning software prototype, the evaluation of the effectiveness of the developed software, and the evaluation of the usability of the software based on three (3) usability attributes, namely ease of use, ease of learning, and task match. CTCode learning software development is based on a combination of the Assure model and the Scrum model. These models were chosen because they have the necessary characteristics to develop learning software. As for the evaluation of effectiveness, the experimental method is used. Evaluation was done on two groups, that is, the treatment group (A) and the control group (B), to see the effectiveness of the developed software.

Usability evaluation of software involves testing the usability attributes based on Eason's model, where evaluating the usability aspects of the developed software. Software usability testing is divided into two levels: user acceptance test and usability test.

Respondents

In this study, the respondents play a crucial role in evaluating the effectiveness of a newly developed software. Evaluation of the respondents' use of CTCode learning will yield the necessary data. CTCode learning is used to enhance computational thinking skills among beginner students through programming courses. Simple random sampling was used to select a total of 60 respondents, who were students enrolled in software engineering courses at a public university. The respondents were divided into two groups: 30 in the treatment group (A) and 30 in the control group (B). These respondents were both male and female and were taking programming courses. For usability evaluation, lecturers from software engineering courses were selected using simple random sampling, resulting in a total of four respondents, including both male and female participants. According to Ocamp et al. (2018), the quantity of respondents does not need to be large, as the quality of results from group discussions is not strongly correlated with the number of respondents.

Instrument

The effectiveness of the learning software we developed was evaluated using a combination of pre-tests and post-tests. Both group A (treatment) and group B (control) took the tests, and the questions were the same for both groups. The pre-test was conducted during the weekly test for that semester, while the post-test was carried out during a programming course. The researcher requested permission from the lecturer to use the teaching session to conduct the test. The test consisted of 13 questions with three parts: objective questions, subjective questions, and code output. The validity of the content was verified by experts. For the usability evaluation, we used testing scripts and checklists as the main instruments. These tools were used to test the prototypes of the modules.

Procedures

The evaluation of effectiveness will be conducted with two groups. The treatment group (A) will use the new software, while the control group (B) will use the existing software. Students will be asked to use the software and complete activities that range in difficulty from easy to hard. Upon completing the software, students will need to answer a programming course question paper that is based on their understanding gained through the use of the software. The marks obtained by the students on the question paper will be recorded to assess the effectiveness of the software.

Regarding usability evaluation, a checklist will be used throughout the learning software usability evaluation process to gather information about the interface of the new learning software. The checklist will be based on the classification of consumerism attributes used in consumerism research.

FINDINGS

This section will discuss and present the research findings obtained through the evaluation of the effectiveness and usability of the CTCode learning software.

CTCode Software's Effectiveness

Based on Table 1, the differences in mean and standard deviation between the pre-test and post-test scores for groups A and B. The pre-test scores were obtained from students during their first programming course test of the semester, while the post-test scores were collected after the students used the specified software. It is important to note that the same students who took the pre-test also participated in the post-test. According to the calculations, the mean score for the treatment group on the pre-test was 50.40, whereas the mean score for the control group was 42.20. For the calculation of standard deviation, the treatment group recorded a standard deviation of 11.30, while the control group had a standard deviation of 9.61. The mean difference in the pre-test scores between Group A and Group B was 8.20. This mean difference reflects the extent of students' understanding of programming before they used the software. This is connected to constructivist theory, which underpins the software activities that were developed.

Additionally, the mean and standard deviation were calculated after the use of the CTCode software compared to existing software. The mean post-test score for the treatment group using the developed software was 74.83, while the mean post-test score for the control group using existing software was 65.13. The mean difference between these two groups was 9.70. This indicates a significant difference in mean scores between the treatment and control groups. The treatment group, which used the CTCode software that incorporates computational thinking, metacognition, and problem-solving skills in each activity, showed improved understanding of programming learning.

Table 1: Mean Score and Standard Deviation

Group	Test	N	Mean	Standard Deviation
A = treatment	Pre-Test	30	50.40	11.30
	Post-Test	30	74.83	14.30
B = control	Pre-Test	30	42.20	9.61
	Post-Test	30	65.13	9.67

Based on Table 1, the mean increased from the pre-test to the post-test for the treatment group to 24.43, while for the control group, the mean increased slightly less to 22.93.

Table 2: Mean Difference of t-Test Scores for Treatment Group

Group	Test	N	Mean	Standard Deviation	t	Significant
A = treatment	Pre-Test	30	50.40	11.30	-6.608	.000
	Post-Test	30	74.83	14.30		

In Table 2, the t-test was used to examine the difference in mean pre-test and post-test scores for the treatment groups. The test result revealed a t-value of -6.608 and a significant p-value (2-tailed) of 0.000. This indicates a significant difference between the mean pre-test and post-test scores following the students' exposure to the CTCode learning software.

Table 3: Mean Difference of t-Test Score for the Control Group

Group	Test	N	Mean	Standard Deviation	t	Significant
B = control	Pre-Test	30	42.20	9.61	-8.371	.000
	Post-Test	30	65.13	9.67		

In the control group, Table 3 indicates a difference in the average scores of the variable before and after the test. The obtained t value is -8.371, and the significance value (2-tailed) is 0.000. This signifies a significant contrast in the mean scores before and after the test, following the use of the current software.

Table 4: Mean Difference of t-Test Score between Treatment Group and Control Group

Group	Test	N	Mean	Standard Deviation	t	Significant
A = treatment	Post-Test	30	74.83	14.30	3.531	0.001
B = control	Post-Test	30	65.13	9.67		

Based on Table 4, a t-test was conducted to compare the post-test scores of the treatment group and the control group. The results of the independent t-test indicated a t-value of 3.531, with a significance value (2-tailed) of 0.001. This suggests that there is a statistically significant mean difference between the post-test scores of the treatment group and those of the control group. Both groups were given access to learning software, which yielded a positive response. The CTCode software, in particular, featured activities that incorporated computational thinking (CT) elements, metacognition, and problem-solving skills. The findings indicate that CTCode software significantly enhances students' understanding and mastery of programming compared to their performance before using any software.

CTCode Software Interface

This section presents the usability evaluation research findings that focus on three key usability attributes: ease of use, ease of learning, and task match. Table 5 shows the findings for the usability evaluation of the level of ease of use.

Table 5: Checklist Score Value Ease of Use Level

No	Items	1 Very difficult *f*	2 Difficult *f*	3 Sufficient *f*	4 Easy *f*	5 Very easy *f*
1	Is this software easy to use?			1(0.25)	3(0.75)	
2	The software utilizes user-friendly menus and activities.				3(0.75)	1(0.25)
3	The steps for the following activities are facile.					
	(a) Sequencing of game levels				3(0.75)	1(0.25)
	(b) Game tasks at each level				3(0.75)	1(0.25)
	(c) Information and evaluation of task solutions				3(0.75)	1(0.25)
4	Users can quickly remember the steps necessary to use this software.				3(0.75)	1(0.25)
5	The graphic icons used in this software are suitable.			1(0.25)	2(0.50)	1(0.25)
6	The screen layout in this software is appropriate.			1(0.25)	3(0.75)	
7	The screen layout in this software is consistent.				3(0.75)	1(0.25)
8	Navigation between screens is easy.				3(0.75)	1(0.25)
9	The provided information is user-friendly.				1(0.25)	3(0.75)
Average		**0.0%**	**0.0%**	**6.8%**	**68.2%**	**25.0%**

The learning software developed is considered easy to use, with 75% of the study participants evaluating it as easy. This indicates a strong user acceptance of its simplicity. In terms of user-friendly features, 25% of the study participants rated it as very easy, while 75% rated it as easy. This suggests that the developed system incorporates user-friendly elements.

The evaluation includes assessing how the CTCode learning software can be used at each level of the game to complete the activities provided. In terms of sequencing of game levels, 25% of the study participants found it very easy, while another 75% found it easy. Similarly, for game activities at each level, 25% of the participants found it very easy, and 75% found it easy. Providing information and evaluation activity answers was aimed at making it easier for users to review their answers. Upon implementation, 25% of the study participants found this activity very easy, while 75% found it easy. These findings demonstrate the acceptance of the activity steps by the study participants.

After using the CTCode learning software and following the usage steps, 25% of the study participants found this activity very easy, while 75% rated it as easy. This finding indicates that the study participants

do not require a long time to remember the steps for using the software. Graphic icons are used to represent the processes that help users understand the activities being carried out. 25% of the study target assessed the graphic icon as very appropriate, 50% assessed it as appropriate, and another 25% assessed it as adequate. This finding shows that the graphic icon used can be accepted by the target audience of the study.

The suitability of the screen layout is crucial for generating user interest in using the CTCode learning software. In this study, 75% of the participants rated it as easy, while the remaining 25% rated it as adequate. This finding demonstrates that the study's target audience finds the screen layout to be acceptable. Maintaining consistency in the layout can assist users in understanding the uniformity of the screens within the software. For instance, using the same arrangement of icons for each screen can make it easier for users to navigate the software. In this study, 75% of participants rated it as easy, and another 25% rated it as easy. This finding indicates that a consistent layout is provided.

Navigation between screens is essential for every process in the CTCode learning software until the required information is reached. The study revealed that 25% of participants found navigating between screens very easy, while 75% rated it as easy. This indicates that the software's screen navigation is generally acceptable to the study participants. One important feature of this learning software is the information section, where users can refer to understand the activities at each level. In a study, 75% of participants found this section very easy, and the remaining 25% found it easy. This indicates that the provided information was well received by the study participants. Table 6 shows the findings of the study based on the aspects studied to test the level of ease of learning.

Table 6: Checklist Score Value Ease of Learning

No	Items	1 Very difficult *f*	2 Difficult *f*	3 Sufficient *f*	4 Easy *f*	5 Very easy *f*
1	Is this learning software easy to learn?				3(0.75)	1(0.25)
2	Do users require training to use this learning software?			1(0.25)	2(0.50)	1(0.25)
3	Can users easily recall the steps for utilizing this learning software?			1(0.25)	3(0.75)	
4	Does it take a long time to learn how to use this software for learning?				3(0.75)	1(0.25)
5	Does the additional information provided help the user take further action?				3(0.75)	1(0.25)
6	Does using menus and activities help in achieving our goals?			1(0.25)	3(0.75)	
7	Is navigation between screens easy?			1(0.25)	3(0.75)	
Average		**0.0%**	**0.0%**	**14.3%**	**71.4%**	**14.3%**

Regarding the ease of learning the software developed, 25% of the study participants rated it as very easy, while 75% rated it as easy. This indicates that the learning software is user-friendly and easy to learn. Regarding the training requirements for using learning software, 25% of the participants rated it as very necessary (score 5), another 25% rated it as adequate, and 50% rated it as necessary. This indicates that training should be conducted before the participants start using the software.

When utilizing this learning software, the process implementation steps will be repeated. As for the study target, 25% found it adequate to remember the steps, while 75% found it easy. This suggests that the steps for using this software are easy to remember based on the study's target. The developed learning software considers the additional information provided to assist users in identifying the situation or task before proceeding to the next step. In this regard, 25% of the study participants found the additional information to be very helpful for taking further action, and 75% found it to be helpful. This finding demonstrates that the provided additional information aids users when using software or engaging in activities.

A well-organized menu and appropriate activities for each level of the game allow users to access the modules provided. 25% of the study participants found the use of menus and activities to be adequate for module access, while 75% found that it facilitated access. This indicates that utilizing the provided menus and activities assists users in utilizing the learning software. 25% of the study target rated navigation between screens as adequate, while 75% found it easy, indicating that it enables easier module access. Table 7 shows the findings of the study related to testing the task match.

Table 7: Checklist Score Values of The Task Match

No	Items	1 Very difficult *f*	2 Difficult *f*	3 Sufficient *f*	4 Easy *f*	5 Very easy *f*
1	Can this learning software assist with teaching and learning activities?			1(0.25)	2(0.50)	1(0.25)
2	Can this software help novice students learn programming?				3(0.75)	1(0.25)
3	Are the provided activities compatible with the programming course's learning objectives?			1(0.25)	3(0.75)	
4	Can the developed learning software improve computational and metacognitive thinking skills?				3(0.75)	1(0.25)
5	Can this learning software reduce the amount of time students spend learning programming courses?			1(0.25)	2(0.50)	1(0.25)
Average		**0.0%**	**0.0%**	**15%**	**65%**	**20%**

The access aspect of this learning software was rated as helpful by 50% of the study target, adequate by 25%, and very helpful by another 25%. This finding demonstrates that the use of learning software significantly aids access as a medium in teaching and learning activities. When using the CTCode learning software to support novice students, 25% of the participants found this aspect very helpful, while 75% rated it as helpful. This suggests that the CTCode learning software is beneficial for novice students learning programming courses.

Regarding the usability of the activities provided to assist users in learning programming courses, 25% of the study participants rated this aspect as adequate, while 75% rated it as important. This finding indicates the significance of the activities provided in facilitating the learning of programming courses, as they align with the topics covered in the course. When it comes to supporting computational thinking (CT) and metacognitive processes, 25% of the study participants rated this aspect as very important, while 75% rated it as important. This finding indicates that the features present in the CTCode learning software are beneficial for developing computational and metacognitive thinking skills.

Regarding the goal of shortening the learning time, 25% of the study participants considered it to be sufficient, 50% considered it to be moderate, and 25% strongly agreed. This indicates that the study participants believe that this learning software is capable of expediting the learning process for programming courses, as activities related to computational and metacognitive thinking skills help students grasp programming concepts more easily.

DISCUSSION

In this section, we will discuss the research findings obtained previously. The discussion will focus on the evaluation of the effectiveness and usability of the CTCode learning software. We will consider attributes such as ease of use, ease of learning, and task match.

CTCode Software Effectiveness Evaluation

The study evaluated software engineering students from a public university who were enrolled in a programming course. The students were divided into two groups: the treatment group, which used the CTCode software, and the control group, which used the existing software. Both groups had to use the assigned software and then answer programming course questions. The evaluation for these two groups was conducted at different times according to their programming course schedule. During the assessment session, students used the provided software and answered quiz questions. The scores from the quiz were recorded for evaluation. The study results indicated a significant difference in mean scores between the post-test of the treatment group and the post-test of the control group. The t value was 3.531, and the two-tailed significance value obtained was 0.001.

These findings suggest that the CTCode software, which incorporates metacognitive techniques and problem-solving strategies, can enhance computational thinking skills in novice students compared to the existing software. The effectiveness of CTCode software in enhancing computational thinking skills can be attributed to its interface design and content, which incorporate metacognitive and problem-solving techniques based on the PDP-CT model in software development. The use of a combination of metacognitive techniques and problem-solving in CTCode software has been shown to enhance the computational thinking skills of novice students.

Evaluation of CTCode Software Usability

The usability level is evaluated after developing the prototype to test ease of use, ease of learning, and task match. The research findings indicate that 68.2% of the research participants found the developed software easy to use and acceptable. The study emphasized the importance of the learning software's usability, particularly in relation to user activities. A well-structured and consistent screen layout was identified as critical to facilitating the software's use. 71.4% of the study's participants found the software easy to learn. The study suggests that training in the use of the software is essential, as the level of understanding varies among respondents, which impacts the ease of learning the software. The findings indicate that the learning software developed meets 65% of the study's requirements. The study shows that the software can be used to improve computational skills among novice students.

CONCLUSION

The development of educational software that integrates game-based elements requires a systematic design process to ensure alignment with pedagogical objectives and learner characteristics. In this study, the creation of CTCode was guided by the Scrum and ASSURE models, which provided a structured

framework for both design and implementation. The evaluation results demonstrate that CTCode is effective in enhancing computational thinking skills among novice learners, as evidenced by the significant improvement in test performance compared to students who used existing software.

Furthermore, the incorporation of activities such as quizzes, drag-and-drop, and mix-and-match tasks proved to be particularly valuable in fostering engagement, metacognitive awareness, and problem-solving strategies. These features not only supported the acquisition of computational thinking but also promoted a more interactive and enjoyable learning experience. The usability assessment also highlighted the software's capacity to provide an intuitive, accessible, and structured learning environment, making it suitable for integration into programming education.

Overall, the findings affirm that CTCode can serve as a practical and impactful learning tool for programming courses, particularly for beginners who often struggle with abstract concepts. By combining effective pedagogy with gamified learning activities, CTCode has the potential to contribute to more meaningful learning outcomes, sustain student motivation, and support the broader aim of cultivating computational thinking as a fundamental skill in the digital era. Future research may further enhance CTCode by expanding its content coverage, incorporating adaptive learning features, and testing its applicability across diverse student populations and educational contexts.

ACKNOWLEDGMENTS

This work was supported by the Computing and Meta-Technology, Universiti Pendidikan Sultan Idris, Perak, Malaysia.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Nurul Najwa Nabihah: Collecting Data, Investigation, and Writing. **Nor Hasbiah Ubaidullah:** Supervision and Reviewing. **Teddie Darmizal:** Reviewing.

DECLARATION OF GENERATIVE AI

During the preparation of this work, the authors used ChatGPT to enhance the clarity of the writing. After using ChatGPT, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

DATA AVAILABILITY STATEMENT

Data available on request from the authors.

REFERENCES

Agbo, F. J., Oyelere, S. S., Suhonen, J., & Laine, T. H. (2021). Co-design of mini games for learning computational thinking in an online environment. *Education and Information Technologies, 26*(5), 5815-5849. https://doi.org/10.1007/s10639-021-10515-1

Akben, N. (2020). Effects of the problem-posing approach on students' problem-solving skills and metacognitive awareness in science education. *Research in Science Education, 50*(3), 1143-1165. https://doi.org/10.1007/s11165-018-9726-7.

Athanasios, C., & Stylianos, M. (2023). Gamification in education. *Encyclopedia, 3*(4), 1223-1243. https://doi.org/10.3390/encyclopedia3040089

Azman, N. S., Rahmatullah, B., Tamrin, K. F., & Qahtan, Y. M. (2024). A preliminary study on tech-based health advisory online system: The case study of UPSI computing students. *AIP Conference Proceedings, 2750, 040006*. https://doi.org/10.1063/5.0148931

Chan, S.-W., Looi, C.-K., Ho, W. K., & Kim, M. S. (2023). Tools and approaches for integrating computational thinking and mathematics: A scoping review of current empirical studies. *Journal of Educational Computing Research, 60*(8),1–45. https://doi.org/10.1177/07356331221098793

Johan, E. J. (2021, November 18). *Rangka kerja reka bentuk tugasan makmal bahasa pengaturcaraan berorientasi objek berasaskan teknologi persuasive* (Doctoral thesis, Universiti Kebangsaan Malaysia). PTSL Digital UKM. https://ptsldigital.ukm.my/jspui/handle/123456789/773296

Grover, S., & Pea, R. (2023). CT+CS: Computational thinking and computer science in schools. *Journal of Educational Computing Research, 61*(1), 54–72.

López-Forniés, I. (2021). Game-based learning and assessment of creative challenges through artefact development. In L. Daniela (Ed.), *Smart pedagogy of game-based learning* (pp. 3–21). Springer. https://doi.org/10.1007/978-3-030-76986-4_1

Jalal, N., Lechen, Z., Linda, M., & Eva, N. (2020). Development of computational thinking, digital competence and 21st Century skills when learning programming in K-9. *Education Inquiry, 11*(1), 1-17. https://doi.org/10.1080/20004508.2019.1627844.

Jayne, L. D., & Matt, S. (2022). Supporting thinking about thinking: Examining the metacognition theory-practice gap in higher education. *Springer, 86*, 99-117. https://doi.org/10.1007/s10734-022-00904-x

Juhaina, A., S., Wajeeh, D., Nimer, B. & Otman, J. (2020). Prospective teachers' development of meta-cognitive functions in solving mathematical-based programming problems with Scratch. *Special Issue Simulation and Modelling in Natural Sciences, Economics, Biomedicine and Engineering, 12*(9), 1569. https://doi.org/10.3390/sym12091569

Julie, D., S., Amanda, J. S., & John, D. (2021). Fostering metacognition to support student learning and performance. *CBE-Life Science Education, 20*(2). https://doi.org/10.1187/cbe.20-12-0289

Maria, S., Hakan, L., & Susane, K. S. (2022). Design for learning programming. *Nordic Studies in Science Education, 18*(1), 6-22. https://doi.org/10.5617/nordina.8251

Mohammed, B. (2023). Importance of metacognitive strategies in enhancing reading comprehension Skills. *Journal of Education in Black Sea Region, 8*(2). https://doi.org/10.31578/jebs.v8i2.291

Mohd Jalaluddin, N. F., & Said, C. S. (2022). Kesan metodologi TRIZ terhadap kemahiran penyelesaian masalah dalam kalangan pelajar Komputeran: Effects of TRIZ Methodology on problem solving skills among Computing students. *Journal of ICT in Education, 9*(1), 61-76. https://doi.org/10.37134/jictie.vol9.1.6.2022

Nurul Dayana, M. D., & Maizatul Hayati, M. Y. (2020). Keberkesanan pengaplikasian pemikiran komputasional dalam pembelajaran berasaskan permainan (PbP) bagi topik operasi asas darab tahun tiga. *International Journal of Education, Psychology and Counselling, 5*(12), 112–122. https://doi.org/10.35631/IJEPC.5350012

Nor Hasbiah, U., Jamilah, H., Suliana, S., & Zulkifley, M. (2020). Enhancing novice computational thinking skills through teaching and learning programming with problem-solving and metacognitive techniques. *Journal of Xidian University, 14*(6), 1967-1979. https://doi.org/10.37896/jxu14.5/493

Nor Hasbiah, U., Jamilah, H., Suliana, S., & Zulkifley, M. (2021). Discovering the role of problem-solving and discussion techniques in the teaching programming environment to improve students' computational thinking skills. *International Journal of Information and Education Technology,11*(12), 615-623. https://doi.org/10.18178/ijiet.2021.11.12.1572

NorBaya, M., R., & Effendi, M., E. M., M. (2023). Coding and computational thinking learning for vocational students: Issues and challenges. *International Journal of Academic Research in Business and Social Science, 13*(9), 112-121. https://doi.org/10.6007/IJARBSS/v13-i9/17766

Norlizawaty, B., & Kamisah, O. (2021). Kemahiran pemikiran komputasional: perkembangan, kepentingan dan pengintegrasian. *International Conference on Business Studies and Education.* https://www.icbe.my/wp-content/uploads/2021/04/Norlizawaty-ICBE3-4.pdf

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21 century skills when learning programming in K-9. *Education Inquiry 11*(1), 1–17. https://doi.org/10.1080/20004508.2019.1627844

Piumi, P., Geetha, T., Supunmali, A., Rangana, P., & Buddhika, C. (2021). A systematic mapping of introductory programming languages for novice learner. *IEEE Access, 9*, 88121-88136. https://doi.org/10.1109/ACCESS.2021.3089560

Riyadi., Triana, J., Syarifah., & Puput, N. (2021). Profile of students' problem-solving skills viewed from Polya's four-steps approach and elementary school students. *European Journal of Educational Research, 10*(4), 1625-1638. https://doi.org/10.12973/eu-jer.10.4.1625

Samantha, P. A., & Ronel, D. P. (2022). Supporting student engagement through the gamification of learning activities: A design-based research approach. *Technology, Knowledge and Learning, 27*, 119-138. https://doi.org/10.1007/s10758-021-09500-x

Ssemugenyi, F. (2022). Trapped at the crossroads: Does problem-based learning make a difference? The moderating role of traditional mode of instruction. *Cogent Education, 9*(1), 1–18. https://doi.org/10.1080/2331186X.2022.2068398

Siti Nordianah, H., H., Hasnul, H., I., Abrizah, I., & Mudiana, M. (2020). Assessment of using Ez-Prog: An easy color schematic model for programming problem solving. *ASEAN Journal of Teaching and Learning in Higher Education, 12*(1), 31-41. https://journalarticle.ukm.my/15424/

Sohail, I. M., Roy, M., Abir, A. S., Jasiya, J., Rim, A. N., & Ragad, M. T. (2021). Enhancing problem-solving skills of novice programmers in an introductory programming course. *Computer Application Engineering Education, 30,* 174-194. https://doi.org/10.1002/cae.22450

Sudin, I. A. A., Rahmatullah, B., Abdullah, M. F. W., Tamrin, K. F., Khairudin, M., & Yahya, S. R. (2022). Kajian tinjauan literatur sistematik terhadap pendedahan pelajar universiti kepada percetakan 3D sebagai persediaan ke industri: A systematic literature review study on university students’ exposure to 3D printing as preparation for industry. *Journal of ICT in Education, 9*(1), 48–60. https://doi.org/10.37134/jictie.vol9.1.5.2022

Unal, C., & Er, B. (2020). Effect of using metacognitive strategies to enhance programming performance. *Information in Education, 19*(2), 181-200. https://doi.org/10.15388/infedu.2020.09

Wang, C. C. (2022). The influence of code.org on computational thinking and learning attitude in block-based programming education. *ICEEL '22: Proceedings of the 6th International Conference on Education and E-Learning,* 235-241. https://doi.org/10.1145/3578837.3578871

Youngjun. K & Seong-Won Kim. (2020). An analysis of pre-service teachers' learning process in programming learning. *International Journal on Advanced Science, Engineering and Information Technology, 10* (1), 58-69. https://doi.org/10.18517/ijaseit.10.1.5723

Zamzuri, M. S. M., Rahmatullah, B., Purnama, S., & Dheyab, O. A. (2024). Development of food sharing IT platform for university community. *AIP Conference Proceedings, 2750, 060002*. https://doi.org/10.1063/5.0148929

Zolkipli, N. Z., Rahmatullah, B., Samuri, S. M., Árva, V., & Pranoto, Y. K. S. (2023). ‘Leave no one behind’: A systematic literature review on game-based learning courseware for preschool children with learning disabilities. *Southeast Asia Early Childhood Journal, 12*(1), 79–97. https://doi.org/10.37134/saecj.vol12.1.7.2023