Journal of Digital Education and Learning Engineering ปีที่ 1 ฉบับที่ 1 (เดือน มกราคม – มีนาคม 2568) หน้า 41-57
Journal of Digital Education and Learning Engineering, Volume 1 No. 1 (January – March 2025) Page 41-57
บทความวิจัย (Research Article)

Journal of Digital Education and Learning Engineering
ดำเนินการวารสารโดย สมาคมการศึกษาดิจิทัลและวิศวกรรมการเรียนรู้

Using the Thinking Process Model Technique to Enhance Students' Computer Programming Skills

Kannika Daungcharone & Krittawaya Thongkoo*

College of Arts, Media and Technology, Chiang Mai University, Thailand

Received: August 18, 2024 Revised: September 17, 2024 Accepted: September 22, 2024

Abstract

The foundation of any technological advancement relies on the work of a programmer. Proficiency in computer programming is indispensable for advancing technology in various domains and is a critical asset for fostering economic expansion and national advancement. However, learning computer programming is not easy. Many students think this subject is complex and challenging to understand, leading to dropouts due to difficulty in analyzing and designing algorithms for program development. This research aims to transform the programming classroom into a new environment using the LTDS to enhance students' programming skills and boost their learning achievements. The LTDS system, a collaborative platform, will encourage students to learn how to design programs using flowchart diagrams in an interactive manner, fostering collaborative learning among classmates. To assess the effectiveness of the new learning approach, we primarily evaluate the student's learning achievement and programming skills across different learning environments and genders. Based on the experimental findings, there is a statistically significant difference in the learning outcomes between students in different learning environments. Those utilizing the LTDS exhibit greater academic achievement and programming proficiency than those in traditional classrooms. Furthermore, a statistically significant contrast was observed between male and female students, with male students demonstrating higher levels of academic achievement and input definition skills. The mean value of all skills is higher for students, regardless of their gender, studying via the LTDS than those who learn in the traditional classroom, regardless of their gender. It indicates that the new learning environment aids students in understanding the lesson, analyzing and designing algorithms, and developing programming skills in setting inputs, outputs, and processing. These are the essential foundations that lead to better learning achievement in programming.

Keywords: Programming skills, thinking process model technique, flowchart diagram, algorithm design

*Corresponding author
Email address: krittawaya@gmail.com

Introduction

In the contemporary era, characterized by rapid and continuous technological advancements, it is crucial to recognize that behind every technological innovation lies the expertise of skilled programmers. Computer programming is fundamental to technological development across various sectors and is pivotal in driving economic growth and national progress. Individuals must develop cognitive and meta-cognitive skills to engage in computer programming effectively. It entails acquiring a deep understanding of the syntax and semantics of programming languages and applying these concepts creatively to solve complex problems. Successful programming requires logical reasoning and imaginative thinking, integrating analytical and creative processes to produce effective solutions (Eteng et al., 2022).

Computer programming subjects often see high dropout and failure rates, influenced by complexity, technical nature, potential lack of logical problem-solving skills, and language barriers (Niekerk & Webb, 2016). These results cause low motivation to learn programming, especially among female students, who usually have low motivation and performance in programming, as many studies indicate (Atmatzidou & Demetriadis, 2016; Sun et al., 2022; Master et al., 2023). Educators and students acknowledge this challenge, particularly in higher education settings. Traditional classroom teaching methods typically involve face-to-face interactions and have proven insufficient in fostering effective learning and ensuring student success in programming (Yang et al., 2022; Hera et al., 2022). Numerous researchers have explored strategies to enhance students' programming education and motivation in response to these challenges. One promising approach involves visual learning environments incorporating diagrams, animations, and drag-and-drop applications. Such methods make the learning process more engaging and enjoyable, transforming it from a passive experience into an active, interactive one. This visual and interactive approach supports the development of higher-order cognitive skills by enabling students to grasp complex programming concepts more effectively (Vahldick et al., 2020; Bak et al., 2020; Lin & Weintrop, 2021). Furthermore, integrating information technologies has become essential for improving students' comprehension and motivation. Online tools and computer simulations offer significant advantages, allowing students to practice and apply programming skills in a dynamic and supportive environment. These technological tools facilitate a more comprehensive and practical understanding of programming concepts, addressing many challenges associated with traditional learning methods (Daungcharone et al., 2020).

This research aims to transform the conventional PHP programming learning environment by leveraging technology to enhance students' abilities to design and develop effective program algorithms. The introduction of the Logical-Thinking Diagnosis System (LTDS) supports students within a collaborative learning framework by providing advanced technological tools. Through LTDS, students can collaboratively design program algorithms using flowchart diagrams, applying a structured thinking process model that enhances their logical reasoning and programming skills. This approach not only supports the development of technical proficiency but also fosters collaborative problem-solving and critical thinking, ultimately

Journal of Digital Education and Learning Engineering, 1(1), 41-57
https://doi.org/10.14456/jdele.2025.4

leading to more promoted motivation and effective learning outcomes in computer programming for both male and female students.

■ Research Questions

To ensure the new approach that applies the LTDS as a tool to create a collaborative learning environment to promote students' learning achievement and programming skills between male and female students. Two questions have been formulated.

RQ1: How do different learning environments affect students' learning achievement among different genders?

RQ2: How do different learning environments affect students' programming skills among different genders?

■ Significance and Purposes

1) To establish a new learning environment that enhances the achievement of programming learning.
2) To evaluate the academic achievements of students in different learning environments based on gender differences.
3) To evaluate students' programming skills in different learning environments based on gender differences.

■ Literature Reviews

This research synthesized data from multiple studies that provide evidence for the effectiveness of the thinking process model technique and technology-supported learning in promoting students' programming skills and learning achievement in basic PHP programming.

Computer Programming and Programming Skills

To effectively learn computer programming, students must develop and apply various critical skills, including problem-solving, logical and algorithmic thinking, and critical reasoning (Kiss & Arki, 2017; Topalli & Cagiltay, 2018). These skills are foundational for understanding the technical aspects of programming and the underlying principles that guide the creation of efficient and effective code. Moreover, Bati et al. (2014) emphasized that the challenges inherent in teaching and learning programming are deeply rooted in cognitive activities. These include solving complex problems, clearly representing algorithms, and creating accurate code syntax. These tasks are further complicated by the need to engage in the technical processes of debugging, editing, compiling, and executing code, all of which require a strong mental model of how

the code operates. Traditional methods of teaching programming in conventional classrooms often focus predominantly on the syntax and structure of programming languages. While these elements are essential, this approach may overlook the importance of helping students fully understand the problems they are trying to solve (Oddie et al., 2010). By emphasizing rote learning of syntax over problem comprehension, students may struggle to develop a deeper understanding of programming concepts, leading to difficulties when faced with real-world coding challenges. Furthermore, Becker et al. (2016) noted that mastering the interpretation of compiler error messages is a crucial aspect of learning programming, as it enables students to diagnose and correct errors more effectively, enhancing their overall coding proficiency. Furthermore, it is important to consider the issue of gender differences, as numerous studies indicate that female students typically achieve lower levels in programming than their male counterparts. For instance, a study shows male students have higher self-efficacy and probability of success in programming (Atmatzidou & Demetriadis, 2016). Similarly, Sun et al. (2022) found that male students can develop programming skills faster than females, and females require more training time to reach the same level of programming skills as male students.

In response to these challenges, several researchers have explored innovative methods to boost students' motivation and improve their performance in learning programming. Kalelioglu (2015) argued that employing diverse instructional methodologies and aids, such as diagrams, animations, and drag-and-drop applications, can significantly enhance students' ability to develop higher-order cognitive skills. These tools provide visual and interactive ways to engage with programming concepts, making abstract ideas more tangible and easier to grasp. For instance, using animations to demonstrate how algorithms work can help students visualize the step-by-step process of solving a problem, thereby deepening their understanding. Bati et al. (2014) further proposed that the use of visualization and simulation tools can be particularly effective in demonstrating the execution steps and runtime behavior of a program. By allowing students to see how their code operates in real time, these tools can reduce the anxiety associated with programming and build students' confidence in their coding abilities. Visualization tools can demystify complex programming concepts, making them more accessible and less intimidating, which is crucial for beginners who might otherwise be discouraged by the steep programming learning curve.

Block-based programming languages such as Scratch and mBlock have become increasingly popular in recent years, especially among students and beginners (Sigayret et al., 2022; Daungcharone & Thongkoo, 2022). These languages are designed to help learners develop programming logic and structure without the immediate pressure of mastering syntax. By using a visual, drag-and-drop interface, block-based programming allows students to focus on the logic behind their code rather than getting bogged down by syntax errors. This approach is particularly effective for introducing programming to younger students or those new to the field, as it simplifies the learning process and makes programming more approachable. Moreover, the visual nature of block-based programming enables students to see the relationships between different parts of their code, helping them to understand how complex programs are constructed. The ability to describe code blocks using plain language further aids in comprehension,

Journal of Digital Education and Learning Engineering, 1(1), 41-57
https://doi.org/10.14456/jdele.2025.4

allowing students to relate programming concepts to everyday language and reasoning. As a result, students can build a strong foundation in programming logic and problem-solving skills, which can later be applied to more advanced programming languages and environments. In summary, mastering computer programming requires more than just learning the syntax of a language. It demands developing critical cognitive skills, understanding and solving problems, and innovative tools that make learning more engaging and less intimidating. Educators can create a more supportive and effective learning environment that fosters motivation and achievement in programming education by incorporating diverse instructional methods, visualization tools, and block-based programming languages.

To learn computer programming, students must go beyond theoretical knowledge and develop a deep understanding of input, process, output, and process structures such as sequence, selection, and repetition. To solve problems, students must possess the ability to comprehend and interpret programming syntax, in addition to being able to construct programs. This procedure poses difficulties and irritations that students must overcome through persistence and dedication (Vahldick et al., 2020; Polito & Temperini, 2021). Computer programming necessitates computational and logical thinking to analyze, solve, and design program processes and understand programming language syntax (Roman-Gonzalez et al., 2017). Computational thinking involves breaking down problems into smaller parts, identifying patterns, and creating algorithms. On the other hand, logical thinking involves using systematic reasoning, deduction, and solving problems by following logical steps (Soufan et al., 2023).

Collaborative Learning Approach Promotes Learning Achievement

Collaborative learning is a teaching strategy that actively engages students in educational tasks by emphasizing collective effort and cooperation among peers. This approach encourages students to participate in shared activities and fosters a sense of self-awareness and social development. Working in groups prompt students to reflect on their contributions and behaviors, which enhances their social skills and deepen their understanding of the subject matter. The nature of collaborative learning requires students to engage with the material and with each other, resulting in a more enriched and meaningful learning experience (Zhang et al., 2021; Ye & Zhou, 2022).

In recent years, the evolution of online learning has given rise to various forms of collaborative learning. This modern adaptation allows students to interact, exchange ideas, and build knowledge together, regardless of time or location. Such flexibility is particularly appealing, as it accommodates diverse learning schedules and fosters a learning environment where students can continuously engage with their peers. The absence of location and time barriers in online collaborative learning has made it a preferred method for encouraging student motivation and achieving educational goals. Students can easily share resources, collaborate on projects, and develop critical thinking skills, all within a virtual space that supports continuous learning and interaction (Miguel et al., 2023; Sabarillo et al., 2023). Both traditional classroom settings and online learning platforms can integrate the principles of collaborative learning to create a more inclusive and engaging educational experience. In traditional classrooms, collaborative learning can manifest through group discussions, peer reviews, and cooperative projects, building a sense of community and

shared purpose among students. Similarly, online learning environments can leverage digital tools such as discussion forums, group chats, and collaborative documents to replicate and enhance the group learning experience. By fostering a sense of belonging and encouraging active participation, these methods help maintain student engagement and contribute to a more dynamic and interactive learning process (Pilotti et al., 2017; Courtney et al., 2022). Research has consistently shown that students behaviorally engaged in online collaborative learning tend to perform better academically. Engagement in this context refers to the active participation of students in learning activities, such as contributing to discussions, completing group assignments, and supporting their peers. Studies have found that students who exhibit high levels of behavioral engagement in online collaborative environments are more likely to achieve superior academic results than their less-engaged counterparts. This relationship between engagement and academic performance underscores the effectiveness of collaborative learning in enhancing educational outcomes (Ye & Zhou, 2022).

The characteristics of online collaborative learning, such as its interactive nature, the motivational aspects of working with peers, and the opportunities for collective problem-solving, contribute significantly to successful learning experiences. Students gain knowledge and develop essential skills such as communication, teamwork, and critical thinking, all of which are crucial for academic success and lifelong learning. In conclusion, online collaborative learning stands out as a powerful educational method that not only boosts learning motivation and achievement but also leads to tangible improvements in academic performance and the development of higher-order thinking skills. This approach helps students achieve more effective and lasting educational outcomes (Liao et al., 2019). In addition, collaborative approaches have also been considered; pair programming is a practice that involves two developers collaborating as a single individual on the design, coding, and testing of the same programming task. Studies have demonstrated that this practice is productive and produces code of higher quality than either developer could produce alone, mainly when dealing with novice programmers or challenging programming problems, and it also yields positive effects in educational contexts (Tunga & Tokel, 2018; Hera et al., 2022).

In summary, integrating collaborative learning theories, the thinking process model technique in a flowchart diagram, and problem-based learning represent a new and innovative approach. This approach harnesses the strengths of these theories to create a learning support system for students studying computer design and programming. The system is designed to motivate students and improve their academic achievement, benefiting both female and male students.

■ Methods

This research investigated the impact of implementing the thinking process model technique to support students' programming skills and academic performance. It examined the disparities between students educated using a traditional approach and those taught using a new approach, including

Journal of Digital Education and Learning Engineering, 1(1), 41-57
https://doi.org/10.14456/jdele.2025.4

considering the impact of different genders. The following section provides a thorough explanation of the experiment.

Participants

The research included 110 first-year students who were studying fundamental computer programming subjects. These students were split into two groups using purposive sampling. The first group (students who enrolled in Section 001) learned in a traditional classroom setting, while the second group (students who enrolled in Section 002) learned via the LTDS in the new learning environment. The first group, known as the control group, comprised 55 students. This group comprised 30 males and 25 females taught using the traditional lecture-based learning method. This conventional approach focused on direct instruction, where students received information through lectures and standard educational materials without the integration of interactive or innovative teaching techniques. The second group, the experimental group, also consisted of 55 students. This group included 28 males and 27 females introduced to a new learning approach incorporating the thinking process model technique. The experimental group engaged in this novel educational strategy to enhance their understanding of programming fundamentals through a more interactive and structured process. This technique was designed to foster deeper cognitive engagement and problem-solving skills by employing a systematic approach to thinking and learning. Table 1 represented detailed demographic information about the participants, including their ages and gender distributions.

Table 1.

The Participant's Demographic Characteristics

Name	Option	Frequency	Percentage (%)
Age	18	61	55.45
	19	43	39.09
	20	6	5.45
Gender	Male	58	52.73
	Female	52	47.27

Setting a New Learning Environment with LTDS

This research aims to transform the traditional PHP programming learning environment by integrating advanced technological tools to effectively enhance students' abilities to design and develop program algorithms. The core objective of this research is to leverage technology to facilitate a more interactive and collaborative learning experience, moving beyond conventional teaching methods. Figure 1 provides a visual representation of the Logical-Thinking Diagnosis System (LTDS), a key tool introduced in this research to support students in a collaborative learning setting. With flowchart diagrams, the LTDS enables students to co-create program algorithms. This approach allows students to work together to tackle

programming problems, fostering a collaborative environment where they can collectively devise solutions. In addition to co-designing algorithms, students can engage in real-time communication with their peers

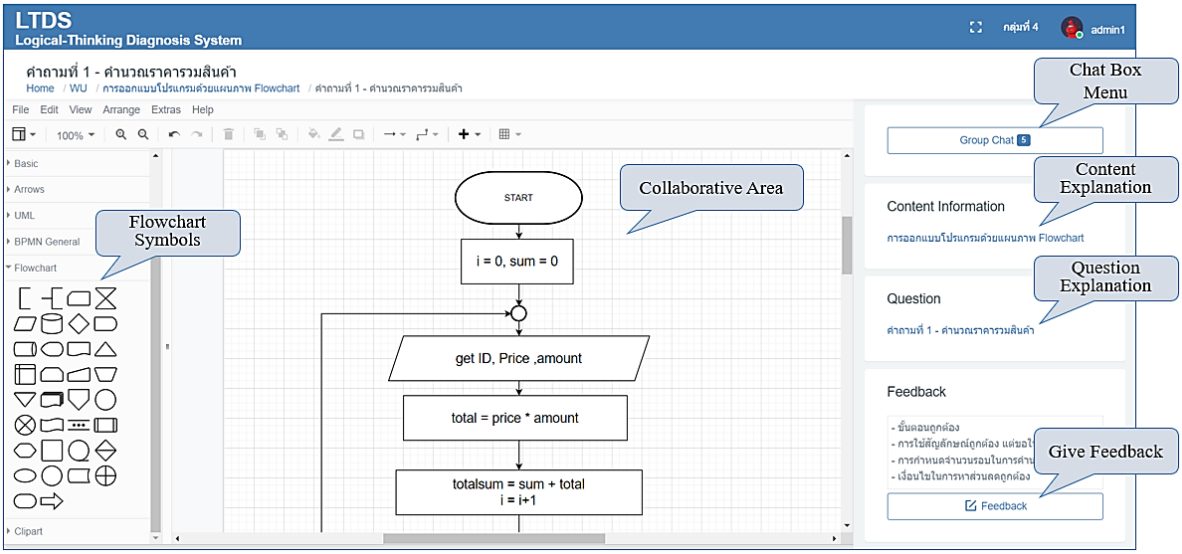

Figure *1.* The LTDS Windows (Daungcharone et al., 2024)

through an integrated chat box feature within the LTDS. This functionality allows them to share ideas, offer suggestions, and provide feedback to one another, further enhancing their collaborative efforts. The LTDS encourages students to participate actively in learning rather than passively receiving information by facilitating open communication and teamwork. The LTDS aims to shift the learning paradigm from traditional lecture-based instruction to a more hands-on, practice-orientated approach. Instead of simply listening to lectures and attempting to understand programming concepts independently, students will use the LTDS to apply their knowledge in practical scenarios. This method promotes experiential learning, where students gain a deeper understanding of programming through active participation and collaborative problem-solving. Overall, integrating the LTDS into the PHP programming curriculum significantly advances teaching methodologies. This research seeks to enhance students' programming skills, improve their problem-solving abilities, and provide a more dynamic and effective educational experience by emphasizing interactive learning and collaborative engagement.

Instruments and Data Collection

Two measurement tools were developed to gather relevant data. The first set of tools consisted of a pre-test and a post-test, both meticulously designed to assess students' learning achievements in PHP programming. These tests evaluated students' understanding of key concepts and skills related to PHP programming design and development. Each test comprises 20 multiple-choice questions, strategically covering essential topics such as algorithms, sequence, selection, and repetition structures. These components are fundamental to understanding and applying PHP programming principles. The pre-test was

Journal of Digital Education and Learning Engineering, 1(1), 41-57
https://doi.org/10.14456/jdele.2025.4

administered at the beginning of the research to establish a baseline of students' initial knowledge and skills. Following the instructional period, the post-test was given to measure any changes in students' learning outcomes and improvements in their programming proficiency. Although the post-test's content may vary slightly from the pre-test's content, it aims to maintain the same level of complexity and coverage. This approach ensures that the assessment remains relevant and effectively measures students' understanding progression while consistently evaluating their grasp of the core programming concepts. Lecturers with extensive experience teaching computer programming subjects reviewed and modified the pre-test and post-test. Their expertise ensured that the tests accurately reflected the learning objectives and provided a reliable measure of students' achievements. It was reliable, with a KR-20 value of 0.69 for the pre-test and 0.70 for the post-test, indicating good internal consistency.

Another measurement tool is rubric scoring, which evaluates the student's programming skills from experimental assignments. The assignment question covers all basic PHP programming, including designing an algorithm by using the structure of sequence, selection, and repetition. The rubric scoring was applied from Joseph's programmer competency matrix (http://sijinjoseph. com/programmer-competency-matrix/) and Daungcharone et al. (2020), including three dimensions, as shown in Table 2. Define input (DI) refers to how students understand the program's required data. Define output (DO) refers to how students determine what data to display after program processing. Create expression (CE) refers to a student's ability to construct PHP expressions for problem-solving. We have categorized the skill scores for the assessment into a range from 0 to 3, each representing different proficiency levels in the subject matter. These scores thoroughly

Table 2.

The Rubric Scoring for Basic PHP Programming

Dimensions	Level 0 Need to improve	Level 1 Low skill	Level 2 Intermediate skill	Level 3 High skill
Define input (DV)	- Lack of problem understanding - Unable to define input	- Not fully understanding the problem - Achieve 50% of the defined input	- Understand the problem - Achieve 100% of the defined input	- Understand the problem - Achieve 100% of defining input - Define input with reasonable order
Define output (DO)	- Lack of problem understanding - Unable to define output	- Not fully understanding the problem - Achieve 50% of show output	- Understand the problem - Achieve 100% of show output	- Understand the problem - Achieve 100% of output - Show output with reasonable order

Dimensions	Level 0 Need to improve	Level 1 Low skill	Level 2 Intermediate skill	Level 3 High skill
Create expression (CE)	- Lack of problem understanding - Unable to create expression	- Not fully understanding the problem - Achieve 50% of create expression	- Understand the problem - Achieve 100% of create expression	- Understand the problem - Achieve 100% of creating expression - Create expression with reasonable order

assess students' abilities and direct their future growth. A score of 0 indicates that students have not yet acquired sufficient knowledge or skills in the assessed area. This rating indicates a fundamental need for improvement and suggests that students would benefit from additional instruction and support to build a foundational understanding of the material. A score of 1 reflects a low level of proficiency. Students with this score have some grasp of the basics but require significant additional practice and study to enhance their skills. This level indicates that while students have started engaging with the material, they have not yet developed the competence to tackle more complex problems effectively. A score of 2 denotes intermediate proficiency. Students achieving this score can solve problems and demonstrate a reasonable level of understanding, but their skills are still developing. They can address the challenges presented but would benefit from further practice to refine their abilities and achieve greater proficiency. A score of 3 represents high proficiency. Students who receive this score have demonstrated the ability to solve all problems comprehensively and correctly. This score indicates that they possess a strong understanding of the material and can apply their skills effectively to address all aspects of the assessment. These scores are designed to offer a nuanced view of students' skill levels, assisting educators and students in pinpointing areas for improvement and acknowledging their learning achievements.

Experimental Procedures

To evaluate the new learning approach to promote student learning achievement and programming skills. The experiment procedures were designed in three main stages, as shown in Figure 2.

Firstly, participants are required to do the pre-test to evaluate their previous PHP programming knowledge in 20 minutes. Secondly, participants were categorized into two groups: the control group and the experimental group. The control group was set to learn basic PHP programming using the traditional approach, in which teachers played a major role. The experimental group was set to learn basic PHP programming in the new learning approach that uses the thinking process model technique as the flowchart (LTDS) to promote students' algorithm design by letting students learn and try to solve problems by themselves with their classmates and teacher play as mentor give them the suggestions. Finally, participants

are required to do the post-test in 20 minutes to evaluate their learning progress after learning via the traditional approach and the new learning approach.

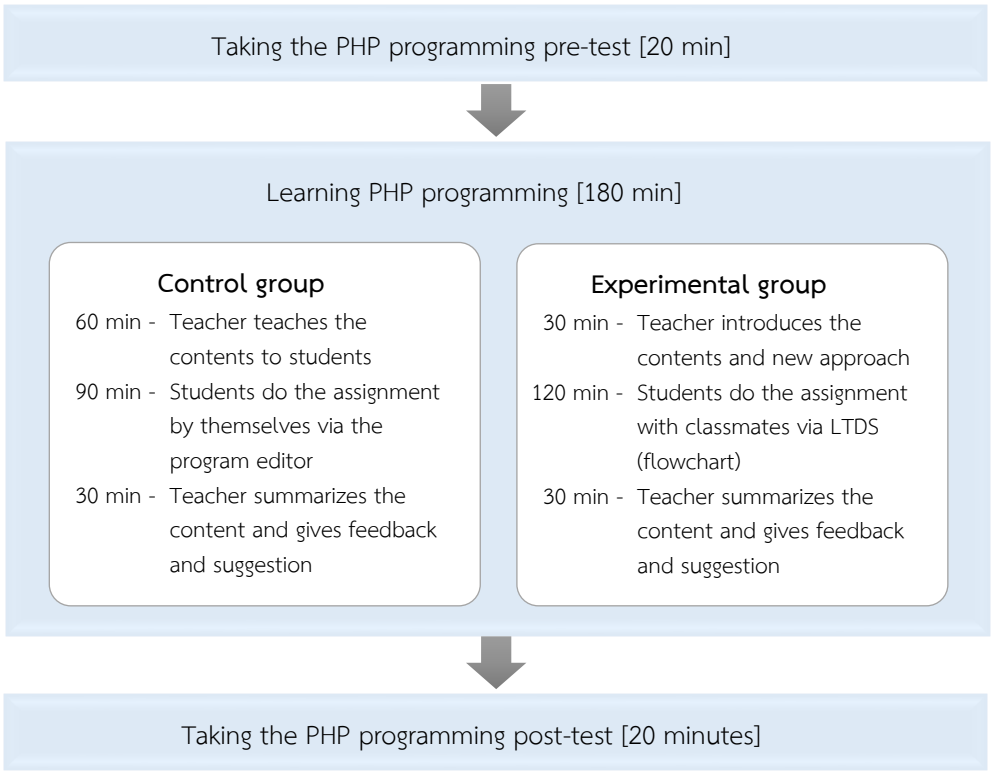

Figure *2.* Experimental Procedures

■ Results and Discussion

According to the research questions, this research examined students' programming skills and learning achievement in different learning approaches, different genders, and the relationship between the learning approach and gender. The experimental results are explained below.

Results of Learning Achievement

Two-way ANCOVA was used to examine how two learning approaches affect learning achievement regardless of prior knowledge. The pre-test score is a covariate variable, the learning approach and gender are independent variables, and the post-test score is a dependent variable. The results displayed in Table 3 indicate that the post-test scores are statistically significant between different learning approaches and different genders, with a small effect size (F = 4.505, p = 0.036, η^2 = 0.041; F = 7.725, p = 0.006, η^2 = 0.069, respectively) but not statistically significant between two learning approaches and gender on the student's PHP programming learning achievement (F = 2.308, p = 0.132, η^2 = 0.022). Table 4 presents the descriptive learning achievement data on the gender indifference learning approach. It shows that male students have higher post-test scores in both traditional and LTDS environments (M = 14.00, SD = 1.912; M = 14.14, SD =

2.445, respectively) than female students (*M* = 12.28, *SD* = 2.458; *M* = 13.89, *SD* = 1.908, respectively).

Table 3.

The Results of Two-Way ANCOVA of Students' Learning Achievement

Source	*MS*	$F_{(1,105)}$	η^2
Learning Approach	17.300	4.505*	0.041
Gender	29.669	7.725*	0.069
Learning Approach * Gender	8.864	2.308	0.022

*$p < 0.05$

Table 4.

Descriptive Data of Students' Learning Achievement between Different Learning Approaches

Learning Approach	Gender	*N*	*M*	*SD*
Learning in a traditional environment	Male	30	14.00	1.912
	Female	25	12.28	2.458
Learning in an LTDS environment	Male	28	14.14	2.445
	Female	27	13.89	1.908

Results of Programming Skills

A two-way ANOVA was used to evaluate students' programming skills, including the ability to define input, define output, and create expression. Learning approaches and gender are independent variables, while PHP programming skills are dependent variables. According to information in Table 5, there are no statistically significant differences between the two learning approaches and gender on the student's PHP programming skills in defining input skill and output skill (*F* = 0.025, *p* = 0.875, η^2 = 0.005; *F* = 0.138, *p* = 0.711, η^2 = 0.005, respectively). However, there are statistically significant differences between the two learning approaches and gender on the student's PHP programming skills in creating expressions (*F* = 4.138, *p* = 0.44, η^2 = 0.038) with a small effect size. Furthermore, when considering different learning approaches, there are statistically significant differences in all programming skills, including defining input, defining output, and creating expression (*F* = 4.166, *p* = 0.044, η^2 = 0.038; *F* = 4.907, *p* = 0.029, η^2 = 0.045; *F* = 63.320, *p* = 0.007, η^2 = 0.038, respectively). In terms of different genders, there is a statistically significant difference in defining input (*F* = 3.914, *p* = 0.049, η^2 = 0.036) but not in defining output and creating expression (*F* = 0.041, *p* = 0.840, η^2 = 0.002; *F* = 0.325, *p* = 0.570, η^2 = 0.003, respectively). Table 6 presents the descriptive data of PHP programming skills (including defining input, defining output, and creating expression) on the gender in difference learning approach. Regarding the learning approach, males and females who learn in the LTDS environment score higher in all skills (male: *M* = 2.18, *SD* = 0.670; *M* = 2.68, *SD* = 0.548; *M* = 2.17,

SD = 0.458; female: *M* = 11.93, *SD* = 0.550; *M* = 2.56, *SD* = 0.751; *M* = 2.01, *SD* = 0.386) than those who learn in the traditional environment (male: *M* = 1.90, *SD* = 0.885; *M* = 2.20, *SD* = 1.157; *M* = 1.21, *SD* = 0.549; female: *M* = 1.56, *SD* = 1.044; *M* = 2.24, *SD* = 1.200; *M* = 1.47, *SD* = 0.576). Moreover, regarding gender, males have a higher skill score than females in all programming skills when they learn in the LTDS environment. While learning in the traditional learning environment, males have higher skills than females in the defined input but have lower skills in the defined output and create expression.

Table 5.

The Results of Two-Way ANOVA Of Students' Programming Skills

Programming Skills	Source	*MS*	$F_{(1,105)}$	η^2
Define input	Learning Approach	2.671	4.166*	0.038
	Gender	2,510	3.914*	0.036
	Learning Approach * Gender	0.016	0.025	0.005
Define output	Learning Approach	4.458	4.907*	0.045
	Gender	0.037	0.041	0.002
	Learning Approach * Gender	0.125	0.138	0.005
Create expression	Learning Approach	15.602	63.320*	0.376
	Gender	0.080	0.325	0.003
	Learning Approach * Gender	1.019	4.138*	0.038

*$p < 0.05$

Table 6.

Descriptive Data of Students' Programming Skills between Different Learning Approaches

Programming Skills	Learning Approach	Gender	*N*	*M*	*SD*
Define input	Learning in a traditional environment	Male	30	1.90	0.885
		Female	25	1.56	1.044
	Learning in an LTDS environment	Male	28	2.18	0.670
		Female	27	1.93	0.550
Define output	Learning in a traditional environment	Male	30	2.20	1.157
		Female	25	2.24	1.200
	Learning in an LTDS environment	Male	28	2.68	0.548
		Female	27	2.56	0.751
Create expression	Learning in a traditional environment	Male	30	1.21	0.549
		Female	25	1.47	0.576
	Learning in an LTDS environment	Male	28	2.17	0.458
		Female	27	2.01	0.386

Discussion

This research examines a new learning method that turns the conventional classroom into a collaborative learning space. This method integrates the effectiveness of the thinking process model technique with a flowchart diagram to improve the computer programming skills and learning outcomes of first-year students in basic PHP programming. The primary objective is to evaluate male and female students' learning achievement and programming skills in diverse classroom settings (traditional and LTDS learning environments).

To address RQ1, the two-way ANCOVA results of the pre-test and post-test indicate that both the learning approach and gender significantly influence students' learning achievement. However, there is no significant interaction between the learning approach and gender, indicating that the learning approach's impact on achievement does not differ significantly between genders, as shown in Tables 3 and 4. In addition, to address RQ2, the analysis of students' programming skills shows that the learning approach significantly influences all three programming skills: defining input, defining output, and creating expressions. Gender has a notable, though comparatively minor, impact on defining input but not on defining output or creating expressions. The relationship between learning approach and gender is typically not statistically significant, except in the case of creating expressions, where it indicates a minor interaction effect, as shown in Table 5 and Table 6.

The experimental results demonstrate that the innovative learning approach effectively transforms a traditional classroom setting into a dynamic and engaging learning environment. This approach significantly enhances students' learning achievements and programming skills by integrating the thinking process model technique, represented through flowchart diagrams, with online collaborative learning. The research reveals several key insights. Firstly, using the thinking process model technique in the form of flowcharts plays a crucial role in improving students' logical thinking abilities. Flowcharts serve as visual representations of algorithms, which help organize and structure information. This graphical depiction aids students in understanding complex programming concepts more clearly and enhances their grasp of programming logic (Bak et al., 2020; Lin & Weintrop, 2021; Sigayret et al., 2022; Daungcharone & Thongkoo, 2022). By providing a structured approach to problem-solving, flowcharts make it easier for students to visualize and manage the steps involved in programming tasks. Secondly, incorporating an online collaborative learning environment complements the thinking process model technique by allowing students to engage in learning activities collaboratively. This online platform removes location and time constraints, enabling students to participate actively in learning regardless of their physical setting. Through collaborative efforts, students engage in meaningful interactions, reflect on their learning experiences, and develop social skills. This active participation fosters a deeper and more comprehensive learning experience, promoting greater engagement and understanding (Miguel et al., 2023; Sabarillo et al., 2023). The hypothesis emerging from this research suggests that the new learning approach enhances students' engagement in mastering basic PHP programming and alleviates any anxiety related to programming challenges. By fostering a more supportive and interactive learning environment, this approach aims to enhance academic performance in both male

and female students, lowering learning barriers and boosting overall success (Tunga & Tokel, 2018; Hera et al., 2022).

■ Conclusion

According to the research findings, integrating online collaborative learning with the thinking process model has numerous benefits. This powerful blend of educational strategies significantly enhances students' learning experiences. By facilitating peer interactions, online collaborative learning platforms play a key role in fostering idea exchange and collaborative problem-solving that can substantially improve learning achievement. Furthermore, the thinking process model enables students to analyze complex problems and design the program's algorithm in a visual form. This method can effectively enhance students' programming skills and learning achievements.

However, certain areas require additional investigation and development. Future research should explore integrating additional features, such as a collaborative coding menu that enables students to code together, observe real-time errors, and engage in joint debugging. This feature would offer immediate feedback and promote a more interactive learning environment. Additionally, there is a need to focus on enhancing programming skills among female students to ensure that they achieve outcomes comparable to those of their male peers.

■ References

Atmatzidou, S. & Demetriadis, S. (2016). Advancing students' computation thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems, 75,* 661-670.

Bak, N., Chang, B. M., & Choi, K. (2020). Smart Block: A visual block language and its programming environment for IoT. *Journal of Computer Languages, 60,* 1–19.

Bati, T. B., Gelderblom, H., & Biljon, J. V. (2014). A blended learning approach for teaching computer programming: design for large classes in Sub-Saharan Africa. *Computer Science Education, 24(1)*, 71–99.

Becker, B. A., Glanville, G., Iwashima, R., McDonnell, C., Goslin, K., & Mooney, C. (2016). Effective compiler error message enhancement for novice programming students. *Computer Science Education, 2(2),* 148–175.

Courtney, M., Costley, J., Baldwin, M., Lee, K., & Fanguy, M. (2022). Individual versus collaborative note-taking: Results of quasi-experimental study on student note completeness, test performance, and academic writing. *The Internet and Higher Education, 55*, 1–14.

Daungcharoen, K., Panjaburee, P., & Thongkoo, K. (2020). Implementation of mobile game-transformed lecture based approach to promoting C programming language learning. *International Journal of Learning and Organisation, 14(2),* 236-254.

Daungcharone, K., & Thongkoo, K. (2022). The effects of thinking process model technique on logical thinking skills influencing programming achievement. *7th International Conference on Digital Arts, Media and Technology and 5th ECTI Northern Section Conference on Electrical, Electronic Computer and Telecommunications Engineering (NCON),* 137–141.

Daungcharone, K., Thongkoo, K., & Panjaburee, P. (2024). Integrating Thinking Process Model Technique with Online Collaborative Learning to Promote Programming Logical Thinking. *International Journal of Information and Education Technology, 14(3)*, 502-509.

Eteng, I., Akpotuzor, S., Akinola, S. O., & Agbonlahor, I. (2022). A review on effective approach to teaching computer programming to undergraduates in developing countries. *Scientific African, 16,* 1-18.

Hera, D. P., Zanoni, M. B., Sigman, M., & Calera, C. I. (2022). Peer tutoring of computer programming increases exploratory behavior in children. *Journal of Experimental Child Psychology, 216,* 1-18.

Kalelioglu, F. (2015). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior, 52*, 200–210.

Kiss, G. & Arki, Z. (2017). The influence of game-based programming education on the algorithmic thinking. *Social and Behavioral Sciences, 237*, 613–617.

Liao, C. W., Chen, C.H., & Shih, S. J. (2019). The interactivity of video and collaboration for learning achievement, intrinsic motivation, cognitive load, and behavior patterns in a digital game-based learning environment. *Computers & Education, 133*, 43–55.

Lin, Y. & Weintrop, D. (2021). The landscape of block-based programming: characteristics of block-based environments and how they support the transition to text-based programming. *Journal of Computer Languages, 67*, 1–18.

Master, A., Tang, D., Forsythe, D., Alexander, T. M., Cheryan, S., & Meltzoff, A. N. (2023). Gender equity and motivational readiness for computational thinking in early childhood. *Early Childhood Research Quarterly, 64,* 242-254.

Miguel, J. P. M., Blas, C. S. D., Rodriguez, F. A., & Sipols, A. E. G. (2023). Collaborative learning in management subjects to university students: A multi-level research to identify group profile, engagement and academic performance. *The International Journal of Management Education, 21*, 1–15.

Niekerk, J. V. & Webb, P. (2016). The effectiveness of brain-compatible blended learning material in the teaching of programming logic. *Computers & Education, 103,* 16-27.

Oddie, A., Hazlewood, P., Blakeway, S., & Whitfield, A. (2010). Introductory problem solving and programming: robotics versus traditional approache. *Inovation in Teaching and Learning in Information and Computer Sciences, 9(2)*, 1–11.

Pilotti, M., Anderson, S., Hardy, P., Murphy, P., & Vincent, P. (2017). Factors related to cognitive, emotional, and behavioral engagement in the online asynchronous classroom. *International Journal of Teaching and Learning in Higher Education, 29*, 145–453.

Polito, G. & Temperini, M. (2021). A gamified web based system for computer programming learning. Computers and Education. *Artificial Intelligence, 2*, 1–13.

Roman-Gonzalez, M., Perez-Gonzalez, J. C., & Jimenez-Fernandez, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the computational thinking test. *Computers in Human Behavior, 72*, 678–69.

Sabarillo, N. S., Sumalinog, D. A. G., & Diazet J. M. S. (2023). Design, implementation, and evaluation of an online flipped classroom with collaborative learning model in an undergraduate chemical engineering course. *Education for Chemical Engineers, 43*, 58–72.

Sigayret, K., Tricot, A., & Blanc, N. (2022). Unplugged or plugged-in programming learning: A comparative experimental study. *Computers & Education, 184*, 1–14.

Soufan, B. A., Bairkdar, B. O., Soufan, E. A., & Samaan, M. (2023). How do college courseand materials affect students' logical thinking of the medical college at Al Baath University in Syria. *Eudacion Medica, 24*, 1–6.

Sun, L., Hu, L., & Zhou, D. (2022). Programming attitudes predict computational thinking: Analysis of differences in gender and programming experience. *Computers & Education, 181,* 1-20.

Topalli, D. & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers & Education, 120*, 64–74.

Tunga, Y., & Tokel, S. T. (2018). The use of pair programming in education: A systematic review. In Paper presented at EDUCCON 2018 Education Conference, Ankara, Turkey.

Vahldick, A., Farah, P. R., Marcelino, M. J., & Mendes, A. J. (2020). A block-based serious game to support introductory computer programming in undergraduate education. *Computers in Human Behavior Report, 2*, 1–12.

Yang, F. C. O., Lai, H. M., & Wang, Y. W. (2022). Effect of augmented reality-based virtual educational robotics on programming students' enjoyment of learning, computational thinking skills, and academic achievement. *Computers & Education, 195*, 1–22.

Ye, J. M. & Zhou, J. (2022). Exploring the relationship between learning sentiments and cognitive processing in online collaborative learning: A network analytic approach. *The Internet and Higher Education, 55*, 1–11.

Zhang, Z., Liu, T., & Lee, C. B. (2021). Language learners' enjoyment and emotion regulation in online collaborative learning. *System, 98*, 1–15.