Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Developing a Teaching Model to Enhance Computational Thinking for Higher Vocational College Students

Yiyan Kong[1], Chowwalit Chookhampaeng[2]

Abstract

The goals of this study were: 1) to study the problems and teaching needs of enhancing computational thinking for higher vocational college students. 2) to develop a teaching model to enhance computational thinking for higher vocational college students. 3) to study the effectiveness of implementing the teaching model to enhance computational thinking for higher vocational college students. The samples were 86 students and 3 teachers in the preliminary survey phase, and 36 students in the implementation phase. Data analysis was done with mean, percentage, standard deviation, content analysis and paired-samples t-test. The research findings showed that the proposed teaching model consisting of eight components and six teaching steps to enhance computational thinking for higher vocational college students. Its significance includes to enhance 4 components of computational thinking: decomposition, pattern recognition, abstraction, algorithm. The experimental students' post-test scores were statistically significantly higher than the pre-test by .05. The results revealed that, the proposed teaching model can effectively enhance the computational thinking for higher vocational college students. Based on the findings of this study, the researchers recommended that teacher combining new technologies such as AI, to provide students with more scaffolding and teaching support and further investigate the effectiveness of this teaching model in enhancing computational thinking in non-programming teaching.

Keywords: *teaching model, computational thinking, higher vocational college student.*

Introduction

Computational thinking (CT) is considered a basic competency required in the 21st century (Voogt et al., 2015). CT enables students to cultivate and develop creative thinking and problem-solving skills, which also enables students to deal efficiently with more complex problems through CT skills (Kong, 2019). Wing (2006) states that rather than being a specific discipline, CT is a way of thinking to solve problems. Therefore, CT skills may enable people to solve more difficult problems (Wing, 2006). computational thinking is an effective tool that can help students and learners develop problem-solving strategies and apply to both their studies as well as everyday life. In an increasingly complicated, digital world, computational thinking can help people tackle a diverse array of challenges in an effective, manageable way. Governments all over the world regard computational thinking as one of the important goals of cultivating students. The Ministry of Education of China (2021) released the 'Information Technology Curriculum Standards for Higher Vocational Education Colleges', which recognized 'computational thinking' as one of the core competencies in the information technology discipline.

There has been broad consensus regarding the close relationship between computational thinking (CT) and programming (Sun,2022). Computational thinking is not coding, but computational thinking may be the outcome of a good planned programming practice (Tedre, 2017). Programming education is an effective mechanism for cultivating students' CT, as indicated by existing studies showing that students' CT can be exercised in the process of participating in programming activities (Angeli & Valanides, 2020; Papadakis et al., 2016; Saxena et al., 2020). There has been a growing interest in using computer programming and coding to enhance students' computational thinking (CT).

Researchers have found through study on the teaching of programming course at Liuzhou Vocational and Technical College, Guangxi, China, that existing teaching models can enhance students' knowledge and skills, but rarely comprehensively promote the development of their computational thinking. This makes it difficult for students to effectively integrate technology with other knowledge, making it difficult to adapt to the rapid updates of technology and the constantly changing demands of professional positions.

[1] Faculty of Education, Mahasarakham University, Mahasarakham, Thailand.
[2] Faculty of Education, Mahasarakham University, Mahasarakham, Thailand.Address: Khamriang Sub-District, Kantarawichai District, Mahasarakham 44150. E-mail: kong_yiyan@163.com (Kong Yiyan), chowwalit.c@msu.ac.th (Corresponding Author)

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Therefore, the present study aims to develop a teaching model to enhance computational thinking for higher vocational college students.

Literature Review

Computational thinking

The term of computational thinking (CT) can be traced back to the 20th century. Papert (1980) stated that CT was one of the skills that required to use a computer and CT could be expressed as the relationship between programs and thinking skills. However, Papert has not expanded the meaning of defining CT and its impact on everyday life (Papert, 2020). Until the 21st century, Wing (2006) put forward a clear definition and discussion on the issue of CT and stated that CT was a kind of ability to solve problems, design systems, and understand human behavior through the basic concepts of computer science. Since then, computational thinking has received extensive attention from the international computer and educational circles, and related research has also been gradually carried out. Guided by Wing's definition and call to action since 2006 (Grover & Pea, 2013), CT is "the thought processes involved in developing problems and their solutions such that an information-processing agent may efficiently carry out the solutions" (Wing, 2008), the concept has been further redefined as "the thinking processes required in creating issues such that their solutions may be expressed as computational steps and algorithms" (Aho, 2012).Ministry of Education of the People's Republic of China (MOE China, 2021) defines computational thinking in the "Information Technology Curriculum Standards for Higher Vocational Education (2021 Edition): computational thinking(CT) is a series of thinking activities generated in the process of forming problem solutions using the thinking methods in the field of computer science.

Regarding the components of computational thinking, Wing (2006) proposed the following characteristics of Computational Thinking (CT) to be considered: abstraction, problem decomposition, problem reformulation, automation, and systematic testing. Anderson (2016) proposes 5 core CT competencies (decomposition, pattern recognition, abstraction, algorithm design, and evaluation) and states that a system can generate a rigorous and efficient solution by following the steps of these competencies. Shute et al. (2017) propose 6 core CT competencies (decomposition, abstraction, algorithm design, debugging, iteration, and generalization) and emphasize the concepts that CT demonstrates through specific skills that enable it to solve problems effectively. The Institute for the Promotion of Teaching Science and Technology (2018) has classified the components of computational thinking into 4 steps as follows: decomposition, pattern recognition, abstraction, algorithm. MOE China (2021) describes the activity of computational thinking as: define problems, abstract features, build models, organize data, solve problems, generalization. This study suggests that, despite inconsistencies in the core concepts and representations proposed by different CT frameworks and models, most of the core concepts defined in these different frameworks are related and similar, in particular, decomposition, pattern recognition, abstraction, and algorithm.1) Decomposition is defined as decomposing a complex problem or system into manageable parts or sub-problems (Anderson, 2016; Shute et al., 2017). 2) Pattern recognition is defined as identifying the regularity and repetition of several sub-problems and finding their patterns and rules (Anderson, 2016). Pattern recognition involves identifying similarities and patterns within and among problems. This component involves recognizing commonalities between different problems or different instances of the same problem and using this knowledge to develop effective solutions. 3) Abstraction is defined as the construction of models based on rules and patterns of data (Anderson, 2016; Shute et al., 2017), which is a way of simplifying a problem or solution by removing extraneous information and focusing only on what is essential to understanding or solving the problem. 4) Algorithm is defined as creating ordered steps to design a solution to a problem (Curzon et al., 2013; Anderson, 2016; Shute et al., 2017). Evaluation is defined as testing and modifying errors to ensure that the steps and processes of an algorithm or a specific model can be executed correctly (Anderson, 2016), and is the same as debugging (Barr & Stephenson, 2011; Shute et al., 2017) and is considered as part of the algorithm (McNicholl, 2018; Harimurti, 2019).

To assess students' Computational thinking, assessment tools are required; these tools vary according to the educational purpose, implementation scope or content, research method, and so forth (Moreno-León

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

et al., 2016). Computational thinking test (CTt) and computational thinking scale (CTs) are suitable for students of all ages and are widely used. Computational thinking test (CTt), such as Bebras tasks (Dagiene & Stupuriene, 2016) and CTT (Román-González et al., 2018), have been applied to examine to what extent learners can transfer CT skills to various situations or problems. CT scales, such as CTS (Korkmaz et al., 2017), can be utilized as a summative assessment tool to evaluate learners' CT level. In addition, design diagrams, flow charts, CT tasks, classroom video analysis, observation notes, etc, have also commonly been used as a formative assessment tool to test learners' CT. At this study, a Computational Thinking Scale (CTs) was developed by researchers to evaluate CT. It was developed by drawing inspiration from the CTs (Korkmaz et al., 2017), which was specifically tailored to the Chinese higher vocational college students. This scale has been created to assess students in four components of computational thinking: Decomposition, Pattern Recognition, Abstraction, and Algorithm.

Strategies for Developing Computational Thinking (CT)

Various research studies have been conducted on the teaching of computational thinking. A review of the literature about How to learn and how to teach computational thinking (Hsu et al., 2018;) found that, about the relationship between subjects and strategies, the most popular strategies in programming courses are project-based learning and problem-based learning. About the relationship between age and strategy, the most commonly used strategy among college students were project-based learning and problem-based learning. Although the application of scaffolding strategies in teaching higher vocational college students is relatively limited, the researchers of this study found that in China, higher vocational college students receive less training in thinking, feel difficult and lack confidence in programming learning, and lack systematic methods to solve difficulties. They hope that teachers can provide a learning framework, and scaffolding strategies can provide assistance to this group of students.

1. Problem-based learning is helping students to set their own learning goals through a problem scene. Students will explore the learning solution by themselves, and report their own learning conclusions and feedback to the team. Problem-based learning is not only used to solve problems, but also to enhance students' understanding of new knowledge through appropriate questions (Wood, 2003).

2. Project-based learning Project-based learning (PBL) is a model that organizes learning around projects. Projects are complex tasks, based on challenging questions or problems, that involve students in design, problem-solving, decision making, or investigative activities; PBL gives students the opportunity to work relatively autonomously over extended periods of time, and culminates in realistic products or presentations (Jones, Rasmussen, & Moffitt, 1997).

3. Scaffolding provides the framework of learning to help the students learn the new knowledge at the beginning. The purpose of scaffolding is to train the students to solve problems independently. Instructional scaffolding is tied to the work of the psychologist Lev Vygotsky, who is well known for several important contributions to educational theory. Vygotsky (1978) coined the term, "zone of proximal development," which is based on a student's current developmental level and potential developmental level. To help a student learn a new task or concept, the teacher targets the student's zone of proximal development and provides support that eventually tapers off as the student grows in knowledge and independence. Vygotsky emphasis on scaffolding (Berger, 2012; Mahn & John-Steiner, 2013), is important in modern constructivist thought. Current interpretations of Vygotsky's ideas emphasize that students should be given complex, difficult, realistic tasks and then be provided enough help to achieve these tasks. This principle is used to support the classroom use of projects, simulations, explorations in the community, and other authentic tasks (Egan, 2008; Levy, 2008; Mahn & John-Steiner, 2013).For example, Basu et al. (2017), Ismail et al. (2010), and Liu et al. (2018) applied mind mapping as scaffolding for college students' programming learning, and showed that students' logical thinking ability, CT concepts, programming ability, innovation competence, and learning performance could all be improved.

4. This study was grounded in Vygotsky's constructivist theory, John Dewey's thinking theory, and Joyce and Weil's teaching model theory, which served as the primary theoretical framework. A novel

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

teaching model was developed by integrating project-based learning, scaffolding strategy, and personalized teaching strategy.

Methodology

Research Objectives

1) To study the problems and teaching needs of enhancing computational thinking for higher vocational college students.

2) To develop a teaching model to enhance computational thinking for higher vocational college students.

3) To study the effectiveness of implementing the teaching model to enhance computational thinking for higher vocational college students.

Research Design

The development of the teaching model in this research proceeded according to three phases.

Phase I, Preliminary survey (R1)

Study the problems and teaching needs of enhancing computational thinking for higher vocational college students. Including 1) investigate the perspectives of higher vocational college students regarding the problems and teaching needs in enhancing computational thinking 2) investigate the perspectives of teachers regarding the problems in enhancing computational thinking for higher vocational college students, as well as their views on developing teaching model that facilitate computational thinking. Using survey research.

Phase II, Develop teaching model (D1)

A teaching model was constructed based on responses to the questionnaire and interview during this phase. The researcher synthesized questionnaire and interview results with a literature survey that included related literature, documents, concepts and theories from the field, constructed a teaching model draft and lesson plans, invite experts to assess the appropriateness of the draft teaching model. These suggestions from the experts were used to refine the tentative model and lesson plans.

Phase III, Implementation (R2)

Implement the teaching model with the target group and study its effectiveness. This involves 1) implementing the developed model with a sample of 36 students for 8 weeks, with 4 class hours per week and 2) study the computational thinking of the students before and after implementing the teaching model. Using action research.

Samples

Samples were selected for two phases of the research.

Phase I, Preliminary survey

In this phase, 1) 86 sophomore and junior students were selected from 178 students majoring in Artificial Intelligence Technology Application at Liuzhou Vocational and Technical College in Guangxi, China， by purposive sampling , for a questionnaire survey. 2) 9 sophomore and junior students majoring in Artificial Intelligence Technology Application at Liuzhou Vocational and Technical College in Guangxi, China, were

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

selected by purposive sampling for interview. 3) 3 professional teachers who have taught programming courses were selected from a pool of 18 professional teachers in the field of artificial intelligence technology application at Liuzhou Vocational and Technical College in Guangxi, China. They were chosen for interview through purposive sampling.

Phase III, Implementation

In this phase, 36 students were selected from 72 first-year students majoring in artificial intelligence technology application at Liuzhou Vocational and Technical College in China，by purposive sampling. They studied at Liuzhou Vocational and Technical College in Guangxi, China, during the second semester of 2023.

Instrument

Phase I: Preliminary Survey

A questionnaire and a semi-structured interview outline were used to investigate the students about the problems and teaching needs of enhancing computational thinking for higher vocational college students. The questionnaire consists of 19 items covering two aspects perspectives on computational thinking and satisfaction with programming instruction. The interview outline listed 16 questions, focus on two aspects of computational thinking and teaching model， in-depth exploration of students' experiences and viewpoints, used open-ended questions to elicit detailed responses and adjusted according to actual situations.

A semi-structured interview outline was used to investigate the perspectives of teachers regarding the problems in enhancing computational thinking for higher vocational college students, as well as their views on developing teaching model that facilitate computational thinking. Additionally, they were asked for their ideas about designing the content and learning activities, and determine the duration for instructional management in each aspect based on their relative significance. The outline comprises 25 questions, focusing on four main aspects: computational thinking, Teaching Method, Teaching Support, and Emotional & Attitude.

They were designed by the researcher, and evaluated by five experts. The IOC values ranged from 0.60-1.00.

Phase II: Develop teaching model

A teaching model draft has been constructed based on the results of the first stage of research, researchers have identified four components of computational thinking that need to be emphasized in teaching: decomposition, pattern recognition, abstraction, and algorithms. The key issues that need to be addressed include: enhancing systematic training of computational thinking, providing personalized support for student learning, and stimulating student interest in learning. The draft teaching model was presented to advisor Checked for accuracy and appropriateness, then made improvements based on advisor's suggestions. An evaluation questionnaire is developed based on the revised draft teaching model and experts are invited to evaluate the appropriateness of the draft teaching model. The evaluative findings were overall at a “High” level of propriety, the mean value = 4.23.

The teaching model to enhance computational thinking for higher vocational college students consisted of 8 components: 1) Theories and Principle 2) Objective 3) Syntax (teaching steps) 4) Social System 5) Principles of Reaction 6) Support System 7) Application 8) Instructional and Nurturant Effects. The third component, syntax, consisted of six steps as follows:

a) Preparation(P). Tasks are prepared before class. Students view and complete knowledge learning and pre-class testing tasks assigned by the teacher on the platform. Teachers analyze the content and design

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

presentation methods; set up learning task sheets, survey questionnaires, and prepare learning materials. Based on students' online grades and survey feedback, teachers analyze the learning situation, determine teaching objectives and key difficulties, and adjust teaching strategies accordingly.

b) Analysis(A). Task analysis includes task understanding, clarify objectives, task decomposition, exploratory learning, and constructing a preliminary plan framework. Teachers utilize videos, case studies, and other resources to assist students in comprehending work tasks, elucidating corresponding knowledge points, skill requirements, and competency expectations. Students are required to clarify learning objectives, confirm assigned tasks, and engage in learning activities guided by task sheets. They can acquire new knowledge through course websites and teacher explanations, reinforce theoretical foundations, and construct preliminary plan frameworks. In the step, to address individual student needs, teacher provide appropriate scaffolding when working with content. The variety of learning task sheets and materials provided by teachers allows students to select scaffolding based on their goals for subsequent learning. By teaching prerequisite content to some students, allowing advanced students to move ahead of the class, or even changing the content for some students based on their individualized education programs. During this phase, students are expected to employ decomposition, pattern recognition, abstraction, and algorithmic thinking to complete tasks.

c) Feedback(F). This step involves knowledge testing, program scheme argumentation and adjustment, and task division adjustment. 1) Students use the questions set by the teacher on the teaching platform in advance to test their knowledge on the course website, self-assess their learning, and receive timely feedback on the test results from the intelligent teaching platform. Students adjust their learning based on the test results. 2) Group members discuss and jointly argue the program, adjust strategies, and select the best solution. Teachers provide precise explanations based on the test results of the learning platform, observe classroom activities, provide feedback on common issues, and offer suggestions tailored to the thinking of each group. 3) Each group member completes the next task according to the division adjustment of labor.

d) Implementation(I). This step mainly involves algorithm design, code writing, program debugging, module integration, and task implementation. Teachers pay attention to student discussions and operations, providing personalized guidance, prompts, or demonstrations in a timely manner. Students divide tasks according to the roles of product managers, development engineers, and test engineers, engaging in imitation exercises, innovative development, or error troubleshooting. They also exchange tasks based on the progress of task completion. During this process, teachers provide diverse learning materials, including videos, images, documents, mind maps, and program demos, to support and guide students with different tasks and progress. Students independently implement algorithm design, code writing, and program debugging. When encountering problems, they can refer to teaching materials at different levels or request teacher support. During the module integration step, multiple collaborators integrate and debug independently completed parts, collaborating to complete the final task. The teaching environment provides students with various equipment, devices, and diverse learning materials, allowing students time and atmosphere for independent thinking and task completion, as well as opportunities for collaborative learning.

e) Evaluation(E). Students take turns presenting their learning outcomes, and conduct self-assessments. Other students provide inter-group evaluations and mutual learning. Teachers provide feedback on task completion, analyze strengths and weaknesses, explain typical problems, and guide discussions. Finally, teachers and students jointly summarize and evaluate the tasks.

f) Improvement (I): This step is typically completed after class. It consists of two parts: summarizing and publishing achievements, and enhancing improvements. 1) Students create mind maps and document summaries of the task completion process and results, record videos, and publish them on the teaching platform. 2) Improve completed tasks, such as performance enhancement, functional improvement, or addition, and complete innovative designs. Teachers provide guidance through online communication tools and select representative works to highlight on the teaching platform.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Phase III, Implementation

In the phase, the teaching model which developed in Phase II is implemented in order to find the effectiveness of the teaching model. There are 2 steps: step 1, implementation teaching according to the instructional model, step 2, pre/post-test 36 students on computational thinking, using Computational Thinking Scale (CTs). Lasting for 8 weeks, with 4 class hours per week.

Computational Thinking Scale (CTs) was developed by drawing inspiration from the CTs (Korkmaz et al., 2017). It is specifically tailored to the Chinese higher vocational college students. This scale has been created to assess students in four components of computational thinking: Decomposition, Pattern Recognition, Abstraction, and Algorithm. The questionnaire is divided into two parts. The first part is basic information, including gender, education background, professional learning foundation, etc. The second part is a self-report questionnaire on computational thinking. According to a 5-point Likert scale, with ratings of strongly agree, agree, not sure, disagree, strongly disagree. There are 16 items in the scale, including decomposition (3 items), pattern recognition (3 items), abstraction (4 items) and algorithm (6 items). Computational Thinking Scale (CTs) was constructed by the researcher and validated by 5 experts to investigate whether each question is consistent with the purpose. Instrument quality is analyzed using the Index of Coherence (IOC) and compared to standards between 0.50-1.00. The consistency index of the scale is between 0.6-1.00, which means the tool can be used.

Data Analysis

1) Results from the questionnaires were analyzed for frequency, mean, percent, and standard deviation.

2) Content analysis was performed on the interview data from students and teachers at Liuzhou Vocational and Technical College. In-depth interviews were analyzed by transcribing the audio recordings and coming to interpretations and conclusions.

3) Pre-test and post-test results were analyzed using a paired-samples t-test in order to compare students' computational thinking before and after implementing the teaching model.

Results

Phase I, Preliminary survey

Problems and teaching needs based on students and teachers' perspectives of enhancing computational thinking for higher vocational college students.

The results of the questionnaire survey:1) Regarding computational thinking, 64.6% of students indicated a lack of understanding of computational thinking, but 100% of them recognized its importance. 89% of students believed it was necessary to understand computational thinking, and 90.2% expressed a desire to enhance their computational thinking. 85.4% of students perceived a close relationship between programming courses and the improvement of computational thinking. 86.6% of students hoped to receive computational thinking training in programming education. However, only 58.5% of students were confident in enhancing their computational thinking, and 67.1% found programming challenging, with an additional 30.5% expressing uncertainty. 65.9% of students believed it was necessary to develop computational thinking teaching models in programming courses, while 32.9% were uncertain. From the analysis of the data above, it can be concluded that the majority of students lack understanding of computational thinking. However, they recognize its importance and usefulness and express a desire to enhance their computational thinking. They agree that programming teaching are closely related to the improvement of computational thinking and hope to develop computational thinking in programming education. More than half of the students expressed interest in programming, but their current challenge is finding programming difficult. Most students believe it is necessary to develop computational thinking teaching models in programming teaching. Approximately 30% of students are undecided about their thoughts on this matter. 2) A satisfaction survey on programming teaching revealed an overall satisfaction

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

score of 4.3, indicating a basic level of satisfaction. The three lowest scores were: individual learning achievement (4.00), student role (4.22), teaching objectives, teaching content, and teaching progress (4.24). Items with higher scores included timely feedback (4.44) and teacher role (4.40). Analysis of the data indicates that students expressed relatively low satisfaction with the learning achievement of the programming teaching. In response to questions about the teaching model, students also expressed lower satisfaction with their role in the classroom, as well as with teaching objectives, teaching content, and teaching progress, which were only rated as basic satisfaction. This guides the need to focus on adjustments to syntax and social systems when developing teaching models. While students showed relatively higher satisfaction with response time to feedback and teacher roles, neither surpassed a score of 4.5. Therefore, there is still room for improvement in these areas when developing teaching models. The details are shown in Table 1, Table 2.

Student Interview Results: Students' satisfaction with the pace of teaching was not high, but their specific needs varied. Among the 9 students interviewed, 3 expressed, "The usual pace of teaching is a bit fast for me. Some content cannot be followed, and I haven't mastered it before moving on to the next section. I hope the teacher can slow down the pace a bit." 2 students mentioned, "The teaching schedule could be more compact. I usually have a lot of spare time after completing tasks." In interviews regarding teaching content, it was also found that while some students felt the content was difficulty was, others had the opposite view.

Teacher Interview Results: Teachers have a relatively vague understanding of computational thinking, with teaching primarily focused on imparting knowledge and skills, neglecting training in thinking. Some students often wait for teacher explanations, finding it difficult to engage in active thinking. However, teachers do not have sufficient time to provide detailed answers to each student's questions. Therefore, there is a need to develop teaching materials at different levels to assist students in learning at their own pace. It is necessary to design teaching activities that ensure both independent thinking and collaborative discussion among students, enabling them to adjust their learning through various means.

Summary of Survey and Interview Results:1) Problems Identified: a) Students' computational thinking needs improvement, including various aspects such as decomposition, pattern recognition, abstraction, and algorithm. b) Students' confidence in learning needs improvement.2) Needs for Teaching Models: a) Systematic training in computational thinking, including targeted training in various aspects of computational thinking and training in overall problem-solving abilities. b) Personalized support for student learning. c) Transformation of students' roles in the classroom. d) Assurance of feedback stage in teaching steps.

Table 1: Results of the questionnaire survey (regarding computational thinking)

Question	Answer	Number of students	Ratio%
1.Do you know about computational thinking?	don't know	53	64.6%
	know	29	35.4%
2.Do you think computational thinking is important	yes	70	85.4%
	not sure	12	14.6%
3.Do you think enhancing computational thinking is useful for your ?	yes	66	80.5%
	not sure	16	19.5%
4.Do you think it is necessary to understand computational thinking	yes	73	89.0%
	not sure	9	11.0%
5.Do you hope to enhance your computational thinking?	yes	74	90.2%
	not sure	8	9.8%
6.Do you think there is a close relationship between	yes	70	85.4%

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

programming courses and enhancing computational thinking?	not sure	12	14.6%
7.Do you hope to receive training in computational thinking in programming learning?	yes	71	86.6%
	not sure	10	12.2%
	no	1	1.2%
8.Do you have confidence in enhancing your computational thinking?	yes	48	58.5%
	not sure	28	34.1%
	no	6	7.3%
9.Do you interest in programming?	yes	46	56.1%
	not sure	34	41.5%
	no	2	2.4%
10.Do you find programming difficult?	yes	55	67.1%
	not sure	25	30.5%
	no	2	2.4%
11.Do you think it is necessary to develop a teaching model of computational thinking in programming courses	yes	54	65.9%
	not sure	27	32.9%
	no	1	1.2%

Table 2 : Results of the questionnaire survey (regarding programming teaching)

Question	Mean
12.Are you satisfied with the teaching objectives in the programming course?	4.244
13.Are you satisfied with the teaching content in the programming course?	4.244
14.Are you satisfied with the teaching steps in the programming course?	4.293
15.Are you satisfied with the teaching schedule?	4.244
16.Are you satisfied with the role of a teacher in programming courses?	4.402
17.Are you satisfied with your role in programming courses?	4.220
18.Are you satisfied with the relationship between the programming course teacher and the students inside and outside the classroom?	4.378
19.Are you satisfied with the response time for feedback received during programming classes?	4.439
20.Are you satisfied with the learning support you received in the programming course?	4.366
21.Are you satisfied with the learning environment of the programming course?	4.268
22.Are you satisfied with the evaluation method of programming achievement?	4.354
23.Are you satisfied with your programming achievement?	4.000

Phase II, Develop teaching model

The development of the teaching model to consisted of 8 components: 1) theories and principle 2) objective 3) syntax (teaching steps) 4) social system 5) principles of reaction 6) support system 7) application 8) instructional and nurturant effects. The third component, syntax, consisted of six steps as follows: Preparation, Analysis, Feedback, Implementation, Evaluation and Improvement. Every component was evaluated by the experts at a "High" level of propriety ($\overline{X}$ =4.23). The details are shown in Figure 1.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

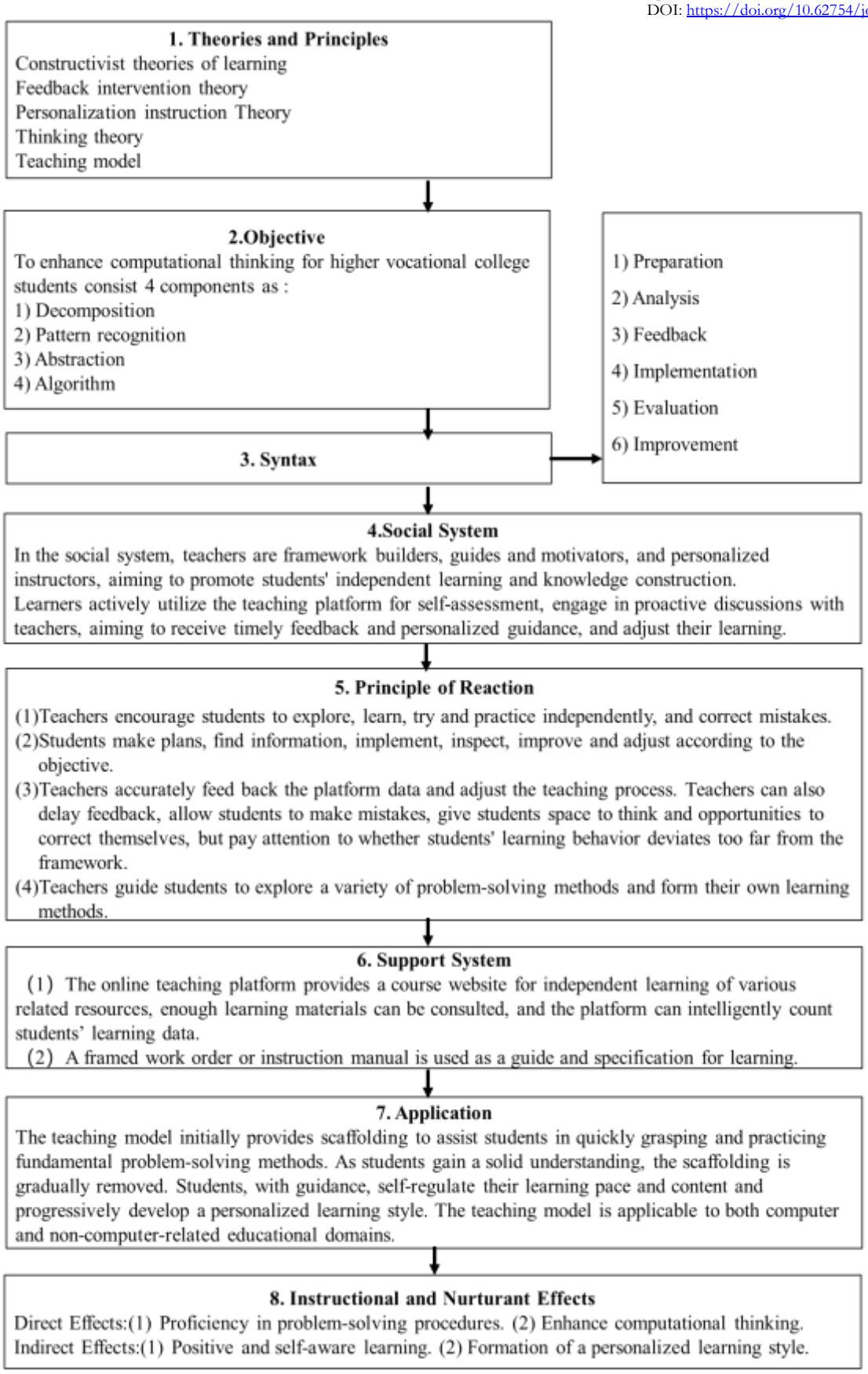

Figure 1 The details of teaching model.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Table 3: The details of syntax

Step	Detail	Teacher Activities	Student Activities
1.Task Preparation (P)	1.Preparation	1) Prepare teaching materials 2) Publish pre-class learning tasks 3) Teaching adjustment	1) Self-study and imitation of operations 2) Complete pre-class testing
2.Task analysis (A)	2.1 Objectives and understanding	1) Assign tasks 1.1) Personal Task 1 (Compulsory) 1.2) Personal Task 2 (optional) 1.3) Group Task 2) Explain the corresponding knowledge, skills, and quality requirements for completing work tasks 3) Communicate with students and help them understand tasks	1) Communicate with teacher and understand the task content 2) Establish the learning objectives that students want to achieve in this lesson
	2.2 Task decomposition	1) Provide scaffolding 2) Provide appropriate guidance on task decomposition and division of labor for groups struggling to carry out their work.	1) Organize group discussions and assign role tasks 2) Conduct group discussions to determine the specific work tasks each member will undertake.
	2.3 Exploratory learning	1) Provide prompting questions 2) Supply learning materials 3) Patrol, personalized guidance, Q&A	1) Self-study 2) Imitate operations 3) Classroom exercises (choose according to progress)
	2.4 Constructing plan framework	Patrol, personalized guidance, Q&A	Construct preliminary plan frameworks for completing the task, such as draw mind maps, flowcharts, list steps, etc.
3. Feedback (F)	3.1 Knowledge testing	1)Knowledge testing. 1.1) publish self-test quizzes on the course website 1.2) publish a learning situation questionnaire survey 1.3) organize knowledge testing game 2) Decide whether to adjust the learning schedule based on the survey results. 3) Teach knowledge, focus on explaining content that students make more errors on and address the most pressing	1) Complete self-test 2) Complete the investigation 3) Participate in activities 4) Listen to the lecture and reflect on it 5) Adjust their learning based on the test results

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Step	Detail	Teacher Activities	Student Activities
		concerns identified in the survey.	
	3.2 Program scheme argumentation and adjustment	Observe classroom activities, provide feedback on common issues, and offer suggestions tailored to the thinking of each group	Group members discuss and jointly argue the program, adjust strategies, and select the best solution.
4. Implementation (I)	4.1 Algorithm design, coding and program debugging	1) Pay attention to student discussions and operations, providing personalized guidance, prompts, or demonstrations in a timely manner. 2) Provide diverse learning materials, including videos, images, documents, mind maps, and program demos, to support and guide students with different tasks and progress.	1) According the divide tasks to engage in imitation exercises, innovative development, or error troubleshooting. 2) They are also allowed to exchange tasks based on the progress of task completion. 3) Independently implement algorithm design, code writing, and program debugging. When encountering problems, they can refer to teaching materials at different levels or request teacher support.
	4.2 Module integration, and task implementation	Personalized guidance , encourage students	Multiple collaborators integrate and debug independently completed parts, collaborating to complete the final task
5. Evaluation(E)	5. Evaluation	1) Explain evaluation methods and rules 2) Organize student evaluations 3) Provide feedback on task completion, analyze strengths and weaknesses, explain typical problems, and guide discussions	1) Publish code and running results 2) Display 3) Group mutual evaluation
6. Improvement(I)	6.1 Summarize and publish results	Online communication and guidance	Create a mind map, summarize the process and results of task completion, record a video, and publish it on the teaching platform.
	6.2 Improvement and expansion	Provide guidance through online communication tools and select representative works to highlight on the teaching platform	Improve completed tasks, such as performance enhancement, functional improvement, or addition, and complete innovative designs.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Phase III, Implementation

The effectiveness of the developed model was tested with its actual implementation. The comparative findings of pre-test and post-test scores found that the experimental group had significantly higher post-test score than the pre-test at .05 level. The details are shown in Table 4, Table 5, Table 6.

Table 4: Results of computational thinking (CT) Pre-test

	N	Mean	Std. Deviation
pre_test_CT	36	3.1771	.62384
pre_Decomposition	36	3.3426	.74530
pre_Pattern_recognition	36	3.4444	.77664
pre_Abstraction	36	3.0694	.61705
pre_Algorithm	36	3.0324	.84967
Valid N (listwise)	36		

Table 5: Results of computational thinking (CT) Post-test

	N	Mean	Std. Deviation
post_test_CT	36	3.5608	.55771
post_Decomposition	36	3.8889	.61205
post_Pattern_recognition	36	3.7593	.70198
post_Abstraction	36	3.4097	.56743
post_Algorithm	36	3.3981	.80008
Valid N (listwise)	36		

Table 6: Results of the comparison between computational thinking pre-test and post-test

		Paired Differences					t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	post_test_CT - pre_test_CT	.38368	.40093	.06682	.24802	.51934	5.742	35	.000
Pair 2	post_Decomposition - pre_Decomposition	.54630	.70891	.11815	.30643	.78616	4.624	35	.000
Pair 3	post_Pattern_recognition - pre_Pattern_recognition	.31481	.52822	.08804	.13609	.49354	3.576	35	.001
Pair 4	post_Abstraction - pre_Abstraction	.34028	.48238	.08040	.17706	.50349	4.233	35	.000
Pair 5	post_Algorithm - pre_Algorithm	.36574	.39805	.06634	.23106	.50042	5.513	35	.000

As shown in the above table, the mean score on the pre-test for computational thinking was 3.17, while the mean score on the post-test was 3.56. The difference was statistically significant ($t=5.742$, $sig=.000$, $p<.05$), indicating that, after implementing the developed teaching model, there was a significant enhancement in computational thinking than before. Each component of computational thinking (decomposition, pattern recognition, abstraction, algorithm) also showed significant enhancement.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Discussion

The objectives of this study were: 1) To study the problems and teaching needs of enhancing computational thinking for higher vocational college students. 2) To develop a teaching model to enhance computational thinking for higher vocational college students. 3) To study the effectiveness of implementing the teaching model to enhance computational thinking for higher vocational college students. Three issues arose to be discussed as follows:

1. The problems in the teaching of computational thinking include: a) Students' computational thinking needs improvement, including various aspects such as decomposition, pattern recognition, abstraction, and algorithm. b) Students' confidence in learning needs improvement. The needs for teaching models include: a) systematic training in computational thinking, including targeted training in various aspects of computational thinking and training in overall problem-solving abilities. b) personalized support for student learning. c) transformation of students' roles in the classroom. d) assurance of feedback stage in teaching steps. These views were derived from investigated perspectives on higher vocational college students and teachers. Consequently, to address these issues, a teaching model should be developed to enhance the computational thinking for higher vocational college students.

2. The developed teaching model to enhance computational thinking for higher vocational college students consisted of 8 components: 1) theories and principle 2) objective 3) syntax (teaching steps) 4) social system 5) principles of reaction 6) support system 7) application 8) instructional and nurturant effects. The third component, syntax, consisted of six steps as follows: Preparation, Analysis, Feedback, Implementation, Evaluation and Improvement. The evaluative findings were scored at a "high" level of propriety. The developed teaching model was based systematically on the approach of Joyce, Weil and Calhoun (2014) whose own model included focus describes the goals and objectives of the model, the principle or basic approach of the model, the details of all the teaching steps, support system, application, instructional and nurturant effects. Arends (2001) concluded that the essential components of a teaching model need to include the goal of the learning activity, which should be based on a proper theoretical approach in developing the learning model. Moreover, the learning process is based on thinking theory (Dewey,1910) , which consists of five logically distinct steps: 1) the occurrence of a difficulty 2) definition of the difficultym3) occurrence of a suggested explanation or possible solution 4) the rational elaboration of an idea 5) corroboration of an idea and formation of a concluding belief (Dewey,1910). The specific teaching process includes 5 steps: first, students are challenged to discover challenging problems in difficult situations to activate their thinking; second, identify the crux of the problem, analyze it, and lay the foundation for subsequent activities; third, propose hypotheses for the problem and analyze the methods for solving it; fourth, compare the problem-solving methods of various hypotheses to find the optimal solution; fifth, solve the problem through practical operation to verify whether the hypothesis is correct, and finally, obtain the result. Many scholars have adjusted the teaching steps based on actual research. This approach is supported by Tang (2022), who developed a teaching model to enhance high school students' computational thinking. The model uses problems as the carrier throughout the entire teaching process, stimulating students' interest in learning through problem orientation. Students enhance their thinking activities in the process of solving problems. She found significant differences between pre-test and post-test data for computational thinking. Su (2021) developed a project-based learning model to improve high school students' computational thinking. His model uses six steps in the learning process: 1) problem aggregation, 2) problem identification, 3) problem abstraction, 4) algorithm design, 5) verification testing, and 6) induction migration. He found significant differences in the post-test for computational thinking between the experimental group and the control group.

3. The findings in implementing the teaching model were that: there is a statistically significant difference between pre-test and post-test at the 0.05 level.

The literature that informed the developed teaching model included such important theories as Piaget's Constructivism (Woolfolk, 1995) focusing on the stimulation by questioning in order to cause new knowledge. Use was made of Vygotsky's approach emphasizing the students' ability to construct knowledge

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

through social interactions with other people, who can support the students who were in lower level of Zone of Proximal Development by scaffolding from teachers and friends through the emphasis on group activity (Slavin,1994). Current interpretations of Vygotsky's ideas emphasize that students should be given complex, difficult, realistic tasks and then be provided enough help to achieve these tasks (rather than being taught little bits of knowledge that are expected someday to build up to complex tasks) (Egan, 2008; Levy, 2008; Mahn & John-Steiner, 2013). Queen (2009) found that the cooperative learning package could enhance the students' higher order thinking as well as problem solving skills. This method was able to develop the students to achieve better learning. They could learn by themselves and cooperate in learning. Personalization instruction is often referred to Differentiated instruction (Doubet & Hockett, 2015; Tomlinson & Moon, 2013), which adapts the content, level, pace, and products of instruction to accommodate the different needs of diverse students in regular classes. The philosophy behind differentiated instruction emphasizes that all children can reach high standards, but some may need tailored assistance to do so (Slavin, 2018). In Anand and Ross's study (1987), fifth and sixth grade students scored significantly higher on solving standard problems and transfer problems after receiving personalized lessons. However, Altun (2012) pointed out that, instructional designers must have a clear understanding of the learning needs and characteristics of each student. Learning paths must then be created that match with individual learners. But it is important that personalization can be expensive and time-consuming if not properly developed and maintained.

Conclusion

This study developed a teaching model to enhance computational thinking for higher vocational college students. The developed model was found to be effective according to specified criteria. 1)Students taught using this model showed significant differences in post-test computational thinking compared to pre-test scores.2) The teaching model has significant differences in enhancing the "decomposition" of computational thinking for higher vocational college students. 3) The teaching model has significant differences in enhancing the "pattern recognition" of computational thinking for higher vocational college students. 4)The teaching model has significant differences in enhancing the "abstract" of computational thinking for higher vocational college students. 5) The teaching model has significant differences in enhancing the "algorithm" of computational thinking for higher vocational college students. Recommendations for future research: 1) teacher combining new technologies such as AI, to provide students with more scaffolding and teaching support, 2) compare the effectiveness of traditional teaching model and the proposed teaching model in enhancing computational thinking for higher vocational college students, 3) further investigate the effectiveness of this teaching model in enhancing computational thinking in non-programming teaching.

Conflict of interest

The authors have not declared any conflict of interests.

Acknowledgment

This study was part of a innovation development project. We would like to thank the Faculty of Education, Mahasarakham University for their financial support. We would like to thank the teachers and students for their participation this study.

References

Arends R (2001). Learning to Teach. (5th ed.). Singapore: McGraw-Hill Higher Education

Aho, A. V. (2012). Computation and computational thinking. The Computer Journal, 55(7), 832–835. https://doi.org/10.1093/comjnl/bxs074.

Anand, P. G., & Ross, S. M. (1987). Using computer-assisted instruction to personalize arithmetic materials for elementary school children. Journal of Educational Psychology, 79(1), 72–78. https://doi.org/10.1037/0022-0663.79.1.72.

Angelia, C., & Valanidesb, N. (2020). Developing young children's computational thinking with educational robotics: An interaction effect between gender and scaffolding strategy. Computers in Human Behavior, 105, 105954. https://doi.org/10.1016/j.chb.2019.03.018.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Anderson, N. D. (2016). A call for computational thinking in undergraduate psychology. Psychology Learning and Teaching, 15(3), 226–234. https://doi.org/10.1177/ 1475725716659252.

Altun, A. (2012). Ontologies for personalization: A new challenge for instructional designers. Procedia - Social and Behavioral Sciences, 64(Complete), 691-698. doi: https://doi.org/10.1016/j.sbspro.2012.11.081.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? ACM Inroads, 2(1), 48–54. https://doi.org/10.1145/1929887.1929905.

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. User Modeling and User-Adapted Interaction, 27(1), 1–49. https://doi.org/10.1007/s11257-017-9187-0.

Berger, K. (2012). The developing person through childhood and adolescence (8th ed.). New York, NY: Worth.

Dagiene, V., & Stupuriene, G. (2016). Bebras-a sustainable community building model for the concept-based learning of informatics and computational thinking. Informatics in Education, 15(1), 25–44. https://doi.org/10.15388/infedu.2016.02.

Dewey, J. (1910). How We Think. Lexington, MA: D.C. Heath and Company.https://doi.org/10.1037/10903-000

Doubet, K. J., & Hockett, J. A. (2015). Differentiation in middle and high school: Strategies to engage all learners. Alexandria, VA: ASCD

Egan, K. (2008). Learning in depth. Educational Leadership, 66(3), 58–64.

George E. Hein (1991) Constructivist Learning Theory.CECA (International Committee of Museum Educators) Conference https://beta.edtechpolicy.org/AAASGW/Session2/const_inquiry_paper.pdf.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43. https://doi.org/10.3102/ 0013189X12463051

Curzon, P. (2013). cs4fn and computational thinking unplugged. In Proceedings of the 8th workshop in primary and secondary computing education (pp. 47–50). ACM. https://doi.org/10.1145/2532748.2611263.

Harimurti, R., Ekohariadi, Munoto, & Asto, B. I. (2019). The concept of computational thinking toward information and communication technology learning. IOP Conference Series Materials Science and Engineering, 535, 012004. doi:10.1088/1757-899X/535/1/012004.

Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers in Education, 126, 296–310. https://doi.org/10.1016/j.compedu.2018.07.004.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). The effects of mind mapping with cooperative learning on programming performance, problem solving skill and metacognitive knowledge among computer science students. Journal of Educational Computing Research, 42(1), 35–61. https://doi.org/10.2190/EC.42.1.b.

Jones, B. F., Rasmussen, C. M., & Moffitt, M. C. (1997). Real-life problem solving: A collaborative approach to interdisciplinary learning. American Psychological Association.

Joyce, B., Weil, M., & Calhoun, E. (2014). Model of Teaching (9th ed.). Boston, MA: Pearson Education.

Kong, S. C., Abelson, H., & Lai, M. (2019). Introduction to computational thinking education. In S. C. Kong, & H. Abelson (Eds.), Computational thinking education (pp. 1–10). Springer, http://dx.doi.org/10.1007/978-981-13-6528-7_1.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). Computers in Human Behavior, 72, 558–569. https://doi.org/10.1016/j.chb.2017.01.005.

Levy, S. (2008). The power of audience. Educational Leadership, 66(3), 75–79.

Liu, Y., Tong, Y., & Yang, Y. (2018). The application of mind mapping into college computer programming teaching. Procedia Computer Science, 129, 66–70. https:// doi.org/10.1016/j.procs.2018.03.047.

Mahn, H., & John-Steiner, V. (2013). Vygotsky and sociocultural approaches to teaching and learning. In W. Reynolds, G. Miller, & I. Weiner (Eds.), Handbook of psychology (Vol. 7, 2nd ed., pp. 117–146.). Hoboken, NJ: Wiley.

McNicholl, R. (2018). Computational thinking using code.org. Hello World, 4, 37.

Moreno-Leon, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it belong in the K-12 curriculum? Journal of Information Technology Education: Research, 15, 283–303. https://doi.org/10.3217/jucs-022-12-1533.

Ministry of Education of the People's Republic of China. (2021). Curriculum Standards for Information Technology in Higher Vocational Education (2021 Edition) [EB/OL]. Retrieved from http://www.moe.gov.cn/srcsite/A07/moe_737/s3876_qt/202104/W020210409562365664467.pdf.

Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2016). Developing fundamental programming concepts and computational thinking with ScratchJr in preschool education: A case study. International Journal of Mobile Learning and Organisation, 10(3), 187. https://doi.org/10.1504/IJMLO.2016.077867.

Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York: Basic Books. https://dl.acm.org/doi/book/10.5555/1095592.

Papert, S. A. (2020). Mindstorms: Children, computers, and powerful ideas. Basic Books.

Queen, S. (2009). Effect of Cooperative Learning and Traditional Strategies on Academic Performance in Middle School Language Arts. Dissertation Abstracts International, 70(04), October. https://scholarworks.waldenu.edu/dissertations/668/.

Román-González, M., Pérez-González, J., Moreno-León, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the computational thinking test. International Journal of Child-Computer Interaction, 18, 47–58. https://doi.org/10.1016/j.ijcci.2018.06.004.

Saxena, A., Lo, C. K., Hew, K. F., & Wong, G. K. W. (2020). Designing unplugged and plugged activities to cultivate computational thinking: An exploratory study in early childhood education. The Asia-Pacific Education Researcher, 29(1), 55–66. https://doi.org/10.1007/s40299-019-00478-w.

Journal of Ecohumanism
2024
Volume: 3, No: 8, pp. 4979 – 4995
ISSN: 2752-6798 (Print) | ISSN 2752-6801 (Online)
https://ecohumanism.co.uk/joe/ecohumanism
DOI: https://doi.org/10.62754/joe.v3i8.5137

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. Educational Research Review, 22, 142–158. https://doi.org/10.1016/j. edurev.2017.09.003

Slavin, R. E. (1994). Educational Psychology: Theory and Practice (4th ed.). Needham Heights, MA: Allyn and Bacon.

Slavin, R. E. (2018). Educational psychology: Theory and practice (12th ed.). New York, NY: Pearson.

Sun (2022). Programming attitudes predict computational thinking: Analysis of differences in gender and programming experience. https://doi.org/10.1016/j.compedu.2022.104457.

Tang (2022). Research on PBL Teaching Model for High School Python Curriculum Based on Computational Thinking 3D Framework. Harbin Normal University.

Tedre, M. (2017). Many paths to computational thinking. Paper presented at the TACCLE 3 final conference, Brussels, Belgium. Retrieved from http://www.taccle3.eu/wp-content/uploads/2017/10/2017_TACCLE3_no_pics.pdf.

Tomlinson, C. A., & Moon, T. R. (2013). Assessment and student success in a differentiated classroom. Alexandria, VA: ASCD.

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35. doi: https://doi.org/10.1145/1118178.1118215.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences, 366, 3717–3725. https://doi.org/10.1098/rsta.2008.0118.

Wing, J. M. (2011), Research Notebook: Computational thinking -what and why? The Link Magazine, 20-23. https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why.

Wood, D. F. (2003). ABC of learning and teaching in medicine: Problem based learning. BMJ: British Medical Journal, 326(7384), 328.

Woolfolk, A. E. (1995). Educational psychology (6th ed.). Boston: Allyn & Bacon.

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. Education and Information Technologies, 20(4), 715–728. https://doi.org/10.1007/s10639-015-9412-6.

Vygotsky, L. S. (1978). Mind in society. (M. Cole, V. John-Steiner, S. Scribner, & E. Souberman, Eds.). Cambridge, MA: Harvard University Press. https://doi.org/10.2307/j.ctvjf9vz4