Elinvo (Electronics, Informatics, and Vocational Education)
Vol 9, No. 1, 2024, pp. 64-75
ISSN 2580-6424 (printed) | ISSN 2477-2399 (online)

Computational Thinking Skills Theorization in The Vocational High School Computer Programming Subject Context

Admaja Dwi Herlambang[1*], Aditya Rachmadi[1]

[1] Universitas Brawijaya, Malang, Indonesia

Article Info

Article history:
Received July 27, 2023
Revised March 22, 2024
Accepted May 21, 2024

Keywords:
Computational thinking skills; computer programming subject; vocational high school; data theorization; grounded theory

Abstract

The purpose of the study is to compile concepts related to computational thinking skills in the context of computer programming subject. The research design used is data theorization (grounded theory) with open coding and axial coding data analysis techniques. Three types of data collection protocols are used: interview protocols, observation protocols, and recording protocols. The characteristics informants used are students who have experience learning computer programming in the vocational high school. The research produced several concepts, namely the problem concept, which then gave rise to the computational thinking skills concept. The problem concept consists of three sub-concepts: the common problem sub-concept, the medium problem sub-concept, and the high problem sub-concept. The concept of computational thinking skills is a set of skills required in the problem-solving process in computer programming. Computational thinking is not just a thought process but an expression of skills that others can observe. In the computer programming subject context, two concepts influence the process of mastering computational thinking skills: the guidance strategies concept and the information sources concept. The concept of success becomes a mediator in developing computational thinking skills. The theory produced in this study can be used as a basis for conducting research with quantitative designs related to deductive proof of the theory.

Corresponding Author:
Email: herlambang@ub.ac.id

INTRODUCTION

Computational thinking (CT) has been the most researched research topic in the last decade [1]. CT is highly recommended to integrate into the instructional process because it can improve the ability of students to cultivate creativity. CT is also recommended to be taught as early as possible to students [2] through the school system because it is believed that students can solve various problems effectively and efficiently using a computational approach [3]. The variety of instructional treatments actualized by teachers and researchers in various subjects to improve students' CT skills is wide [1]. This diversity is triggered because the point of view or paradigm related to the concept of CT has much variety, so the praxis based on these concepts will result in a different instructional operationalization. The diversity of instructional treatments for CT debriefing is also due to the different characteristics of the subject [4].

Currently, there are five subjects when viewed from the scientific literacy point of view, namely science, technology, engineering, art, and math [5]. The current trend's five subjects are combined or integrated into one (transdisciplinary) and termed STEAM [6]. Integration in STEAM aims to equip students related to scientific literacy and creative aspects. Scientific literacy and creative aspects are believed to be the primary abilities of the 21st century that must be provided to students through the school system. Scientific literacy and creative aspects can guide the student to solve problems that are volatility, unpredictable (uncertainty), not simple (complexity), and ambiguous (VUCA) in times of

information explosion (information overload, global data shock, or infobesity) [3], [5], [7]–[9]. The position of CT theory in STEAM is as an order of thinking in problem-solving and developing reasoning.

The postulates put forward by researchers and experts related to CT as a theory are too abstract, so it cannot be considered a standard method that can be directly contextualized into curricular situations in a specific subject. The curricular classification for the sciences, technology, engineering, art, and mathematics, independently and integrated, has different characteristics and cannot use only one point of view of CT theory. Some theorists position CT as the epistemological locus of the computer science subject to understand how computational concepts work through instructions given to computer science problems (CT as a subject) devices [10]. Some theorists position CT as a methodology choosing alternatives to high-level thinking (reasoning) in order to solve any problem outside the computer science subject (non-computer science problem, CT as a method/ general-purpose thinking tool) [11].

Learning computer programming can condition students to get used to thinking in sequence or algorithmic thinking [12]. Computer programming is a commonly used means of familiarizing and visualizing the CT process as a method of reasoning for students [1], [13]–[15]. Computer programming is a tool for generating computational artefacts that have an essential role in helping students conclude abstract reasoning experiences. Computational artefacts make it possible to fuse CT into subjects other than computer science (non-computer science problems). The fusion of CT with other computer science subjects has become easier because computer programming can be taught by two methods, namely unplugged (without computer devices) or on the computer (by using a computer) [16].

In the current Society 5.0 era, individuals are not only required to prioritize reasoning skills in solving problems but are also required to have the ability to work together between individuals or between platforms in order to achieve joint stability in the future (collaborative society) through collaboration-based activities supported by information technology, such as co-construction or co-learning activities [17]–[19]. The ability to collaborate has now become an international educational issue where UNESCO has determined collaboration competency to be one of the critical competencies to achieving Sustainable Development Goals (SDGs) by 2030 [20]. The Government of the Republic of Indonesia has also directed that the facilities and infrastructure provided by educational institutions can support collaborative learning. The ability to work together should be familiarized through an instructional system teachers deliberately design in the school system (collaborative learning, co-learning) [21]. Collaborative learning is an instructional method that allows students to work together to explore a problem or create a project [22], [23]. The goal of collaborative learning is that students can find a more contextual and rich learning experience through many points of view while integrating new information with old information that has been stored in long-term memories rather than finding individual learning experiences . This objective is a postulate of social learning theory. The construction of cognition or acquisition of knowledge content by the individual needs to be supported by self-encouragement and the individual's interaction with his environment to reinforce the contextualization of the knowledge gained in the community/ society [24], [25].

The various study results that researchers and experts have presented can be used as an indication that the diversity of instructional actualization integrating CT through computer programming will continue to occur, and this is a result of the variety of substances of CT theory used by teachers as designers of instructional systems. The theories that experts have reviewed can enrich CT epistemology's characteristics, but they cannot necessarily be suitable for practice in all situations of the learning environment. Segregation of forms of knowledge taught, variety of online or offline modalities, and variety of collaboration within instructional systems have never been specifically discussed regarding the implementation of CT theory, so this will potentially be the basis for the development of CT theory with a new context. Thus, this study aims to theorize the experience data of informants/respondents related to computer programming learning activities that have occurred in the context of online learning modes, collaborative learning strategies and uses procedural forms of knowledge. These contexts can still expand or narrow depending on the study's success in unearthing the data the first time (first cycle).

Table 1 shows that the research designs used to examine CT are diverse: systematic literature review, item development and analysis, experimental design, and basic interpretative study. No researcher has yet used the grounded theory method to develop a new paradigm related to CT theory. The results of research from Tikva & Tambouris [15] have not yet entirely created a theory with a new paradigm related to CT, although at first glance, it looks like it is making a new model or theory. Tikva & Tambouris [15] only reassembled existing theories based on the iterator studies they conducted on

some research results so that the resulting theory is not a new theory But only adjustments or rearrangements to make it more straightforward. The basis of CT theory used by researchers in Table 2.1 is diversity, namely looking at CT as a set of ways or strategies of thinking consisting of decomposition, abstraction, algorithm design, debugging, iteration, and generalization. If CT is viewed as a set of ways of thinking, then contextualizing it across fields will be difficult. For example, training CT in students through computer programming and mathematics subjects will be easier than training CT through subjects such as language or art. The difficulty is because the elements in CT are defined initially based on how computers or computing tools work, so it is effortless to contextualize them in the eyes of computationally based programming, mathematics, or other lessons. However, if people want to apply CT in non-compute subjects, then CT theory derived from the computational paradigm must be studied first or even be compiled paradigm new so that training CT skills students are easier to operationalize in the school system. All the research in Table 1 also does not highlight specific scopes such as how to teach CT skills in online learning, how CT develops in collaborative learning, or whether there needs to be a policy that computer programming is required to be taught to all areas of science as general-thinking tools, or a combination of these three things.

Table 1. Related Works

Researchers	Result	Research Design	Data Sources
Luo, et al. [26]	The integration of CT into the field of mathematics can improve students' problem-solving skills	Basic interpretative study	22 elementary students
Kutay & Oner [27]	CT capabilities can be developed through games	Experimental design	20 elementary students
Kim, et al. [28]	Development of CT-based ICT literacy assessment instruments	Item development and analysis	23,000 elementary and middle school students
Tikva & Tambouris [15]	Conceptual model of CT	Systematic literature review	101 results of research on CT
Lai [29]	Development of competency-based CT capability assessment instruments	Item development and analysis	119 high school students
Shute, et al. [1]	Variety of definitions, teaching, assessment, and CT models	Systematic literature review	70 research results on CT
Flórez, et al. [13]	CT capabilities can be developed through computer programming	Systematic literature review	92 results of research on CT

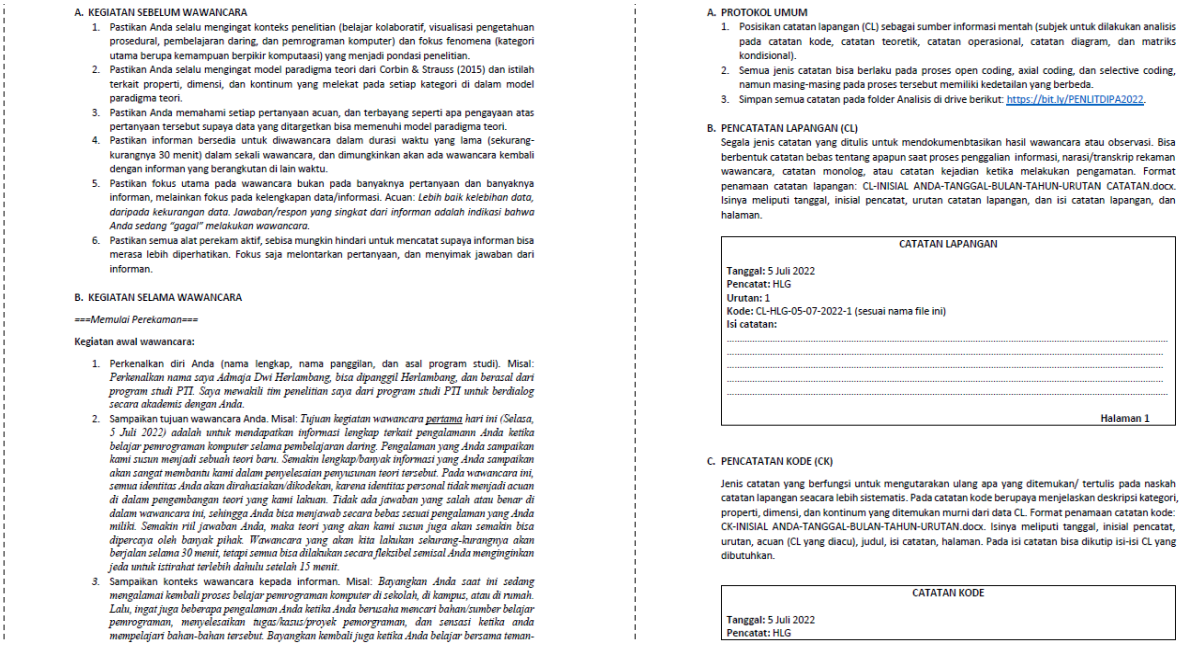
METHODS

The main findings expected from the research are in the form of theoretical formulations that contain labeling of concepts and relationships between concepts based on qualitative data obtained from informants or resource persons and field notes. The theory that will be produced in the study is different from the usual description/narrative, where the labeling in the description is only an overview of the theme without containing concepts (properties and dimensions) and makes no effort to detect interrelationships between themes. The material object theorized in this study is temporarily termed computational thinking (CT) concept with computer programming subject context. The term for the material object and the concepts are temporary because they will be readjusted to the properties and dimensions of the concept found during the analysis process data takes place. Research has a formal object in the form of an inductive approach with qualitative design and a data theorizing method (grounded theory). The primary reference used for the operationalization of research is the procedure of Corbin & Strauss [30].

The research location was carried out in three State Vocational High Schools with Computer Engineering and Informatics Expertise. The three schools are SMKN 3 Malang, SMKN 12 Malang, and SMKN 3 Batu. The three vocational schools are in Malang City, East Java, Indonesia. The selection of research locations in the three vocational schools is because the school allows the production of data under the characteristics of research-based research problems and material objects that have been determined in research. There are several considerations for choosing informants as data sources, namely teachers who teach computer programming subjects and students who have or are currently

following subjects in computer programming. The selection of the school's location and the informant's characteristics was the initial initiation to collecting data in the first cycle. The data collection results in the first cycle will likely affect updating the characteristics and number of research locations and informants in the following data collection cycle.

This study uses three data collection protocols: interview protocols, observation protocols, and recording protocols. The protocol samples are presented in Figure 1. The interviewer will use the interview protocol as a semi-structured guide when communicating orally with the informant. Observers will use the observation protocol to identify what phenomena will enrich the drafting of concepts or categories related to the material object under study. The research team will use the recording protocol to standardize each qualitative analysis result so that the history of adding or subtracting data can be rediscovered when analyzing data. In the first data collection cycle, the content of the interview protocol and observation protocol is determined based on the theoretical sensitivity related to the concept of computational thinking and its initial contexts, such as the computer programming subject. The interview protocol contains a list of questions, while the observation protocol contains a list of what things need to be observed in the form of a list matching the open field. The recording protocol is determined based on the reference from Corbin & Strauss [30], where the reference is used as the basic format of the document for code records, theoretical notes, operational records, diagrams, and conditional matrices.



A. KEGIATAN SEBELUM WAWANCARA

1. Pastikan Anda selalu mengingat konteks penelitian (belajar kolaboratif, visualisasi pengetahuan prosedural, pembelajaran daring, dan pemrograman komputer) dan fokus fenomena (kategori utama berupa kemampuan berpikir komputaasi) yang menjadi pondasi penelitian.
2. Pastikan Anda selalu mengingat model paradigma teori dari Corbin & Strauss (2015) dan istilah terkait properti, dimensi, dan kontinum yang melekat pada setiap kategori di dalam model paradigma teori.
3. Pastikan Anda memahami setiap pertanyaan acuan, dan terbayang seperti apa pengayaan atas pertanyaan tersebut supaya data yang ditargetkan bisa memenuhi model paradigma teori.
4. Pastikan informan bersedia untuk diwawancara dalam durasi waktu yang lama (sekurang-kurangnya 30 menit) dalam sekali wawancara, dan dimungkinkan akan ada wawancara kembali dengan informan yang berangkutan di lain waktu.
5. Pastikan fokus utama pada wawancara bukan pada banyaknya pertanyaan dan banyaknya informan, melainkan fokus pada kelengkapan data/informasi. Acuan: *Lebih baik kelebihan data, daripada kekurangan data. Jawaban/respon yang singkat dari informan adalah indikasi bahwa Anda sedang "gagal" melakukan wawancara.*
6. Pastikan semua alat perekam aktif, sebisa mungkin hindari untuk mencatat supaya informan bisa merasa lebih diperhatikan. Fokus saja melontarkan pertanyaan, dan menyimak jawaban dari informan.

B. KEGIATAN SELAMA WAWANCARA

===Memulai Perekaman===

Kegiatan awal wawancara:

1. Perkenalkan diri Anda (nama lengkap, nama panggilan, dan asal program studi). Misal: *Perkenalkan nama saya Admaja Dwi Herlambang, bisa dipanggil Herlambang, dan berasal dari program studi PTI. Saya mewakili tim penelitian saya dari program studi PTI untuk berdialog secara akademis dengan Anda.*
2. Sampaikan tujuan wawancara Anda. Misal: *Tujuan kegiatan wawancara pertama hari ini (Selasa, 5 Juli 2022) adalah untuk mendapatkan informasi lengkap terkait pengalamann Anda ketika belajar pemrograman komputer selama pembelajaran daring. Pengalaman yang Anda sampaikan kami susun menjadi sebuah teori baru. Semakin lengkap/banyak informasi yang Anda sampaikan akan sangat membantu kami dalam penyelesaian penyusunan teori tersebut. Pada wawancara ini, semua identitas Anda akan dirahasiakan/dikodekan, karena identitas personal tidak menjadi acuan di dalam pengembangan teori yang kami lakuan. Tidak ada jawaban yang salah atau benar di dalam wawancara ini, sehingga Anda bisa menjawab secara bebas sesuai pengalaman yang Anda miliki. Semakin riil jawaban Anda, maka teori yang akan kami susun juga akan semakin bisa dipercaya oleh banyak pihak. Wawancara yang akan kita lakukan sekurang-kurangnya akan berjalan selama 30 menit, tetapi semua bisa dilakukan secara fleksibel semisal Anda menginginkan jeda untuk istirahat terlebih dahulu setelah 15 menit.*
3. Sampaikan konteks wawancara kepada informan. Misal: *Bayangkan Anda saat ini sedang mengalamai kembali proses belajar pemrograman komputer di sekolah, di kampus, atau di rumah. Lalu, ingat juga beberapa pengalaman Anda ketika Anda berusaha mencari bahan/sumber belajar pemrograman, menyelesaikan tugas/kasus/proyek pemorgraman, dan sensasi ketika anda mempelajari bahan-bahan tersebut. Bayangkan kembali juga ketika Anda belajar bersama teman-*

A. PROTOKOL UMUM

1. Posisikan catatan lapangan (CL) sebagai sumber informasi mentah (subjek untuk dilakukan analisis pada catatan kode, catatan teoretik, catatan operasional, catatan diagram, dan matriks kondisional).
2. Semua jenis catatan bisa berlaku pada proses open coding, axial coding, dan selective coding, namun masing-masing pada proses tersebut memiliki kedetailan yang berbeda.
3. Simpan semua catatan pada folder Analisis di drive berikut: https://bit.ly/PENLITDIPA2022.

B. PENCATATAN LAPANGAN (CL)

Segala jenis catatan yang ditulis untuk mendokumenbtasikan hasil wawancara atau observasi. Bisa berbentuk catatan bebas tentang apapun saat proses penggalian informasi, narasi/transkrip rekaman wawancara, catatan monolog, atau catatan kejadian ketika melakukan pengamatan. Format penamaan catatan lapangan: CL-INISIAL ANDA-TANGGAL-BULAN-TAHUN-URUTAN CATATAN.docx. Isinya meliputi tanggal, inisial pencatat, urutan catatan lapangan, dan isi catatan lapangan, dan halaman.

CATATAN LAPANGAN

Tanggal: 5 Juli 2022
Pencatat: HLG
Urutan: 1
Kode: CL-HLG-05-07-2022-1 (sesuai nama file ini)
Isi catatan:

Halaman 1

C. PENCATATAN KODE (CK)

Jenis catatan yang berfungsi untuk mengutarakan ulang apa yang ditemukan/ tertulis pada naskah catatan lapangan seacara lebih sistematis. Pada catatan kode berupaya menjelaskan deskripsi kategori, properti, dimensi, dan kontinum yang ditemukan murni dari data CL. Format penamaan catatan kode: CK-INISIAL ANDA-TANGGAL-BULAN-TAHUN-URUTAN.docx. Isinya meliputi tanggal, inisial pencatat, urutan, acuan (CL yang diacu), judul, isi catatan, halaman. Pada isi catatan bisa dikutip isi-isi CL yang dibutuhkan.

CATATAN KODE

Tanggal: 5 Juli 2022
Pencatat: HLG

Figure 1. Data Collection Protocol Sample (left: interview protocol, right: logging protocol)

The research team used code notes to record the history of labeling, propitiation, and dimensionalization of a concept or category based on the narrative of qualitative data obtained from interviews and observations. The content of the code record is the record code; date of record creation; title/ reference code to interview/observation narratives; the title of the note (in the form of a label); excerpts of interview/observation narratives; and narratives resulting from the extraction of information by the research team (consisting of properties, dimensions, as well as symptoms/ dynamics of the continuum that can be used as material for drafting hypotheses). The research team used theoretical notes to re-narrate the results of the code notes but based on the researchers' theoretical sensitivity and critical thinking. The content of the theoretical record is the code of the note; date of record creation; title/ code reference to code notes; the title of the note (in the form of a label); the narrative of the results of reflection on the code record (consisting of properties, dimensions, and symptoms/ dynamics of the continuum based on the knowledge of the researcher). The theoretical record is directed to subsequent analytical actions, such as retrieving data on the next cycle to validate the theoretical records' narrative through new cases that will be found next. An operational record reviews the theoretical record of the

following required action. The content of the theoretical record is the code of the note; date of record creation; title/ code reference to code and theoretical notes; the title of the note (in the form of a label); narratives related to what should be done at a later stage (for example: what kind of informants are needed, what questions will be asked to the informant, linkages such as what you want to ascertain the related tassels, concept/category labels, properties, dimensions, and dynamics of the continuum). The research team used diagrams to visualize the processes and flow obtained based on the history of code, theoretical, and operational records. Visualization of diagrams, with no standardization, can be used as a benchmark, meaning that it can adjust like anything in shape as long as it can simplify the meaning of the process and the flow of interaction between visual concepts/categories that may get more complex as the notes grow. There is no standardization of conditional matrices that can be used as a benchmark, meaning that they can adjust as they look as long as they can be used to visualize space scope, boundaries, or context, as well as the interconnectedness between concepts/categories found.

All data recorded by the research team were analyzed using the Text Statistic Analyzer Software. This application helps calculate words in a research note and makes it easier for the research team to create labels or concepts. The software display is presented in Figure 2.

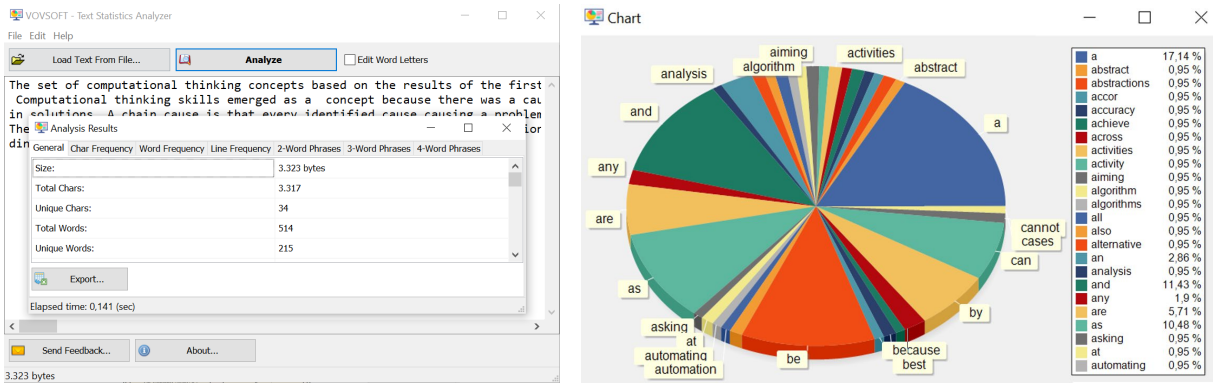


Figure 2. Text Statistic Analyzer Software

RESULT AND DISCUSSION

The Conceptualization Process

Data were collected based on interview protocols, observation protocols, and recording protocols. The informants used were five students from each school. The informant's characteristics have experience following the computer programming learning process. The process of conceptualizing computational thinking is divided into levels: level one, level two, and level three. Level one has a keyword occurrence criterion of more than 50. Level two has a keyword occurrence criterion between 30 and 50. Level three has a keyword occurrence criterion of less than 30. The words included at level one based on the results of the first session of the interview are "problem" (f=70; 70 times appeared in the interview text), "solution" (f=64), "algorithm" (f=63), "problem" (f=60), "programming" (f=60), "skill" (f=55), "automation" (f=51), "abstraction" (f=50), and "decomposition" (f =50). The words represent fundamental technical concepts related to computational thinking.

The words included in level two based on the results of the first session of the interview are "communication" (f=48), "practice" (f= 47), "collaboration" (f=44), "confident" (f=42), "creative" (f=41), "repeating" (f=41), "meaningful" (f=37), "presentation" (f=37), "demo" (f=36), "case" (f=33), and "assignment" (f=30). These words represent the computational thinking development strategy concept in learning. The words included at level three based on the results of the first session of the interview are "media" (f=27), "system" (f=26), "simulation" (f= 24), "book"(f=19), "internet" (f=18), "infrastructure" (f=17), "material" (f=14), "puzzle" (f=1 4), "source" (f=14), "plan" (f=12), "create" (f=11), "flowchart" (f=9), "multimedia" (f=7), "robot" (f=6). These words represent the information source concept required in computational thinking development. The relationship between concepts is presented in Figure 3. The red arrows indicate the causal relationship between concepts. Each concept has sub-concepts. Figure 3 shows that the study found 5 concepts and 15 sub-concepts. The

ISSN 2580-6424 (printed) | ISSN 2477-2399 (online)

Computational Thinking Skills Instructional Concept is the big concept name given by the research team that overshadows the interconnectedness of the five concepts.

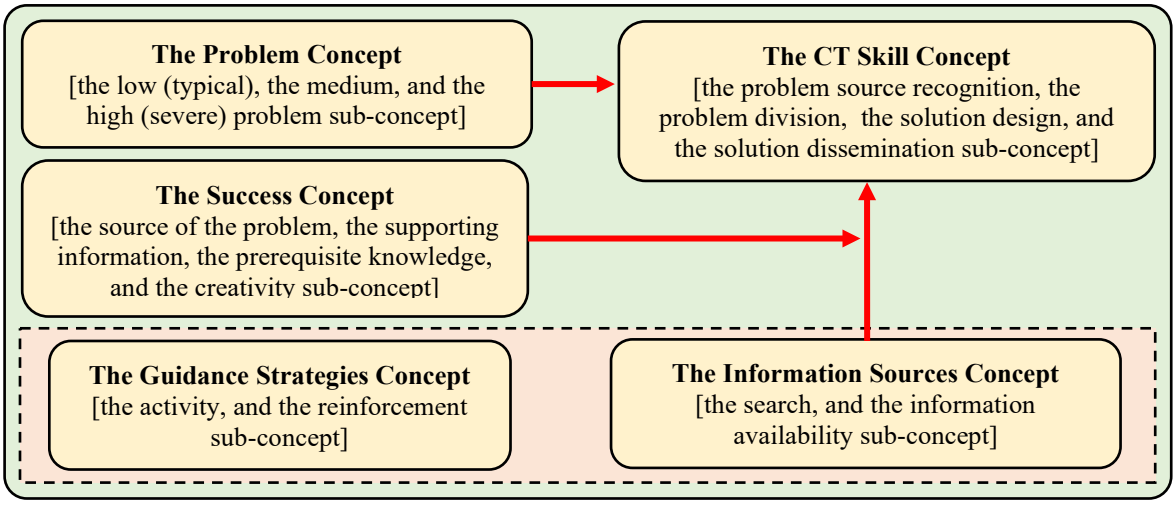


Figure 3. Computational Thinking Skills Instructional Concept Visualization

The Computational Thinking Skills Concept

The set of computational thinking concepts based on the results of the first session of the interview can be described as a set of essential skills involved in the problem-solving process, such as computationally formulating problems, processing data to solve problems, automating solutions with explicit algorithms, and generalizing and transferring problem-solving processes. Computational thinking is not just a thought process but a skill whose processes and outputs can be observed by others. The expression of the output encourages individuals to detect potential abstract problems, decompose the substance of the problem into simple, and design a solution that can be communicated to other people's and repeatedly used to solve the same problem (automation). This process of communication then positions computer programming as one of the best visualization media to monitor how the process of development of computational thinking activity in individuals.

Computational thinking skills emerged as a concept because there was a causal factor in the form of an urgent problem to determine its solution. A problem is any situation that occurs unexpectedly, aiming to prevent something from happening, something that must be resolved, or an unpleasant or unwanted condition that needs to be corrected. In the computer field context, the problem's source must first be identified so that the resulting solution has the accuracy of use. Efforts to find the source of a problem and determine various alternative solutions to problems are often termed problem-solving efforts. The problem as an object of solving the problem has a gradation of complexity: low (typical), medium, and high (severe). A typical problem is described as a problem caused by one cause, and the solution is only one. The medium problem is described as a problem caused by more than one cause, and the solution can be more than one. Severe problems are described as problems caused by chain causes and chain solutions. A chain cause is that every identified cause causing a problem has several previous causes. A chain solution is that any proposed solution cannot directly solve the problem but must be followed by the next solutions in a link.

The results of the analysis of the interview activities of the first session stated that problem-solving skills are at the core of computational thinking skills and are essential to training students as early as possible. This concept shows that computational thinking skills are contextualized as a material (lesson) that must be taught to students through a learning program or system. Computational thinking skills are necessary for problem-solving in computer programming and can also be used to support problem-solving in all subjects. Computational thinking skills can be taught across different subjects, for example, in Mathematics or Languages [31], [32]. Computational thinking skills can be taught to students by asking students: to break down complex problems into more minor problems (decomposition); to recognize patterns (pattern recognition); to identify relevant details to solve problems (abstractions), or to establish rules or instructions to follow to achieve the desired result (algorithm design) according to the cases or problems context raised by the teachers.

ISSN 2580-6424 (printed) | ISSN 2477-2399 (online)

The Guidance Strategy Concept

The development of computational thinking through computer programming is a complex activity that requires students to have high-level thinking skills and involves the teacher's creativity in presenting various problems (cases) with varying levels of difficulty for students to solve. The development of computational thinking through computer programming has problem-solving content and requires students to use logic to encourage the formation of computational thinking skills. Learning is prioritized so that students move or practice in various problem spaces presented by the teacher (inquiry) and problems found by students during the learning process (discovery), so that students can express the knowledge gained in the form of solution designs or prototypes. Variations in the difficulty level of the teacher's problems will increase student motivation in learning.

The development activity of computational thinking through computer programming indicates that students are facilitated to use logic. Students, as learning actors, must be able to diagnose and solve technical problems found during computer programming learning activities. Students' ability to solve technical problems with abstract logic is called "knowledge worker" or "abstract reasoning". Learning activities provide students with various case studies to elaborate on problems and make individual and collective decisions with other students. The goal of teacher measurement of student activity is not on the quantity of programming code produced but is focused on the quality of the algorithms compiled by students to solve problems.

The principle of developing computational thinking through computer programming is the meaningfulness of knowledge in learning so that it can lead students to gain meaningful experiences in learning activities. Indicators of programming learning activities, that is, there are activities to develop skills of analysis, design, code writing, testing, recycling, and decision making. Learning activities that can be practised by teachers so that students get learning practice can be divided into preconception, demonstration, and consolidation activities. Preconception activities are activities in which the teacher is in charge of recognizing the initial understanding possessed by students. Demonstration activities are where teachers demonstrate various prototyping skills and concept prototypes. Students are tasked with adopting various skills demonstrated by the teacher and solving various problems encountered during the learning process. Consolidation activities are activities in which teachers strengthen students' understanding by providing various tasks on computer programming. Analytical tasks and programming code development must be completed in teamwork. Teamwork can trigger students to get used to working in groups (collaboration), conveying or presenting thoughts in the form of solution designs to others, and openness in accepting a wide variety of potential solutions to the same problem.

In the main activity of computer programming instruction, teachers can apply various learning methods. The concept of learning methods learning in the technology subject is the meaningfulness of knowledge in learning (meaningful knowledge that will be recognizable to students) so that it can lead students to gain meaningful experiences meaningful learning experience and develop higher-order thinking skills. The concept of meaningful learning can be obtained through the application of learning methods that can facilitate students to adopt the skills (imitating demonstrated skills) because: most of the knowledge in computer programming learning is procedural or practical knowledge; focusing on the formation of high order thinking skills in students because the application of knowledge gained from learning computer programming requires high reasoning skills; and focuses on building the working character of the software engineering field in students through student involvement in work teams and project management. These three criteria can be obtained from demonstration-based, discussion, and project-based learning methods.

The development of computational thinking through computer programming does not mean that student activity is focused on writing a large amount of programming code and students' proficiency in using software alone. The development process of computational thinking also needs to be supported by latent aspects such as confidence in identifying problems, motivation and perseverance in repeating and trying to solve problems, tolerance for ambiguity, decision-making ability to solve open problems, the ability to communicate, the ability and work with others to achieve common goals or solutions, and relevant assignments. These latent aspects will become obstacles to internalizing computational thinking skills in the learning process if they are not properly facilitated.

One way to focus students' attention on learning computer programming is to form students into a group work environment and ask them to prepare for the technical needs of computer programming together. Working groups can strengthen students' focus at the beginning of learning, improve communication skills, increase self-confidence, increase motivation and knowledge, and speed up the

assessment process [23]. Some limitations to focusing students' attention in learning computer programming, namely the teacher guiding students to form a work team which is then used as a discussion environment in solving problems, the teacher guiding students to prepare simulators to simulate a process, the teacher guiding students through demonstration activities or simulating the process flow of an algorithm, and the teacher explains the learning objectives to be achieved after the learning process.

Another way to arouse student motivation in the initial learning activities is to invite students to analyze a phenomenon related to the programming topic to be studied, provide cases of debugging computer programs, and display visualizations or animations on the programming topic to be studied. Activities that invite students to observe some real cases on the topic of programming can arouse student motivation to participate in learning. Activities to motivate students at the beginning of learning should invite students to analyze concrete things instead of talking about abstract things. Activating students' prior knowledge is mandatory for teachers when teaching computer programming. Teachers can conduct an initial assessment to determine what students have mastered related to the subject matter to be taught through test and non-test methods. Teacher can conduct an initial test (pre-test) on the ability of students to understand the learning material before the core learning activities are carried out [33], [34]. Teacher can give some questions directly to the students to find out the interests that the students have [35].

A good assignment is a task that can strengthen and hone students' skills on an ongoing basis so that a match or familiarization is found with what the software engineering industry needs. The intended task is essential to developing the student's abilities as a coder. The various tasks given by teachers to students must have elements of understanding, maintaining, refactoring, adaptation, and extension to programming codes. Understanding means that the task aims to enable students to understand the structure of the programming code algorithm. The maintaining element means that the task aims to make students able to correct errors that occur in the program code. The refactoring element means that assignments aim to enable students to make decisions to use the most effective and efficient algorithms. The adaptation element means that the task has the goal that students can adapt a programming code to solve a problem or solution. The extension element means that the task aims for students to develop basic programming concepts to produce a completely new product. Assignments related to maintaining and extension elements must be completed in teamwork.

The Information Sources Concept

Teachers need a variety of relevant sources of information so that the process of internalizing computational thinking skills can be channeled to students in computer programming subject. Information that the teacher will use can be sourced from what already exists or self-created by the teacher. Whether existing or self-created, teachers still have to plan the infrastructure so that there is relevance to developing computational thinking skills. The sources of information in question are divided into two groups: the searched information and the provided information. The searched information is any sources of information that are deliberately searched and found and can be used by students to support the learning process so that students can deepen, try something new, or dig up much information according to the learning content. The requirement for the searched information is to have clear relevance to the learning content. There are two types of sources for the searched information: textbooks and the internet. Textbooks can be in the form of print media or electronic media. The internet is also one of the dynamic learning resources supported by the advancement of website technology developments. The internet is dynamized as a learning resource because it can support learning that activates students where students can freely extract information about what is being studied [36]–[38].

The provided information is any resource the teacher can deliberately prepare to facilitate the delivery of information to students. The information provided by the teacher is prioritized in the form of visual media because the substance of learning computational thinking skills involves identifying and solving problems. Visual media is an essential medium widely used in the learning process to form practical and theoretical knowledge [39]–[41]. Five forms of visual media can be used: handouts, visual aids, presentation slides with presentation software, software system, and hardware systems. Handouts contain vital elements that the teacher will deliver in textual form. Visual aids are a board or displays that can synchronize a flowchart. The presentation slide is a show of text, images, sounds, or a combination of them in a package called a slide. System software helps display or simulate steps by step a process interactively and can contain multimedia components. Hardware systems help display or simulate the step-by-step process physically or mechanically, such as robots or puzzles.

ISSN 2580-6424 (printed) | ISSN 2477-2399 (online)

The Success Concept

The concept of success intended in this study is a source of causes or obstacles that result in students having difficulty or making it easier to develop computational thinking skills. There are four sources of difficulty in developing computational thinking skills, namely (1) the vagueness of the process of identifying the source of the problem; (2) irrelevant supporting information; (3) insufficient prerequisite knowledge; (4) the lack of creativity in designing solutions. The vagueness of identifying the source of the problem occurs when the teacher releases a problem without being followed by clear boundaries. The vagueness can lead to a protracted, multi-interpretation, and sometimes deviating process of identifying the source of the problem [42], [43]. The form of the problem can be made deliberately by the teacher (inquiry), or students independently determine their problem (discovery). Whatever the form of initiation, teachers are still required to provide clear boundaries so students can detect whether the problem at hand is a low (typical), medium, or high (severe) problem.

Irrelevant supporting information can result from learning resources or teaching materials that are irrelevant or do not support problem-solving in the development of computational thinking. Whether learning resources or teaching materials, it should be ascertained by the teacher whether the information in it can enlighten students or mislead the problem-solving process done by students. Too little and too much information will make solving the problem protracted and even out of context [42], [44], [45].

Less prerequisite knowledge from students can trigger students to have difficulty in developing computational thinking skills. Teachers must ascertain whether students' prerequisite knowledge can be used for problem-solving. The knowledge at least includes the identification of the problem and the design of the solution to be made. Problem-solving activities in the development of computational thinking skills are included in the higher-order thinking skills, and students should have sufficient conceptual and procedural knowledge.

Students' lack of creativity can also potentially hinder the development of computational thinking skills. One of the hallmarks of computational thinking skills is the compilation of problem-solving sequences that are then proposed into a model, design, or prototype to refer to in decision-making. The contextualization of problems, in reality, differs from the problems presented in the learning program. That is, problems are dynamic, can change at any time, and no single solution is precisely the same from time to time. Thus, students cannot rely solely on previous experience to solve current and future problems, but they must have a boost of creativity to create new proposals, models, or solution designs [46]–[48].

CONCLUSION

The research produced several concepts, namely the problem concept, which then gave rise to the computational thinking skills concept. The problem concept consists of three sub-concepts: the common problem sub-concept, the medium problem sub-concept, and the high problem sub-concept. The concept of computational thinking skills is a set of skills required in the problem-solving process in computer programming, which consists of a problem source recognition sub-concept, a problem division sub-concept, a solution design sub-concept, and a solution dissemination sub-concept. Computational thinking is not just a thought process but an expression of skills that others can observe.

In the computer programming subject context, two concepts influence the process of mastering computational thinking skills: the guidance strategies concept and the information sources concept. The guidance strategy concept consists of the activity sub-concept and the reinforcement sub-concept. The information sources concept consists of the search sub-concept and the information availability sub-concept. The success concept becomes a mediator in developing computational thinking skills. The success concept consists of four sub-concepts: the source of the problem sub-concept, the supporting information sub-concept, the prerequisite knowledge sub-concept, and the creativity sub-concept.

The Computational Thinking Skills Instructional Concept produced in this study can be used as a basis for conducting research with quantitative designs related to deductive proof of the theory or hypothesis. For example, research can be conducted to test in experimental research design whether creativity affects the speed at which new solutions are created for various problems in computer programming. The results of this study are limited to the connection between concepts without finding the connection between the sub-concepts. Further research can also be carried out with grounded theory design using selective coding to find connections between sub-concepts.

REFERENCES

[1] V. J. Shute, C. Sun, and J. Asbell-Clarke, "Demystifying computational thinking," *Educ. Res. Rev.*, vol. 22, pp. 142–158, Nov. 2017, doi: 10.1016/j.edurev.2017.09.003.

[2] M. Wang and Y. F. Wang, "A Study on Computer Teaching Based on Computational Thinking," *Int. J. Emerg. Technol. Learn.*, vol. 11, no. 12, p. 72, Dec. 2016, doi: 10.3991/ijet.v11i12.6069.

[3] M. Mohaghegh and M. Mccauley, "Computational Thinking: The Skill Set of the 21st Century," *Int. J. Comput. Sci. Inf. Technol.*, vol. 7, no. 3, pp. 1524–1530, 2016.

[4] M. S. Khine, "Strategies for Developing Computational Thinking," in *Computational Thinking in the STEM Disciplines*, Cham: Springer International Publishing, 2018, pp. 3–9. doi: 10.1007/978-3-319-93566-9_1.

[5] N. G. Lederman and J. Lederman, "Nature of Scientific Knowledge and Scientific Inquiry," in *Contemporary Trends and Issues in Science Education*, Switzerland: Springer Nature Switzerland AG, 2020, pp. 3–20. doi: 10.1007/978-3-030-57646-2_1.

[6] J. M. Spector, "Education, Training, Competencies, Curricula and Technology," in *Emerging Technologies for STEAM Education*, Cham: Springer International Publishing, 2015, pp. 3–14. doi: 10.1007/978-3-319-02573-5_1.

[7] O. Mack and A. Khare, "Perspectives on a VUCA World," in *Managing in a VUCA World*, Cham: Springer International Publishing, 2016, pp. 3–19. doi: 10.1007/978-3-319-16889-0_1.

[8] L. McIntyre, *Post-Truth*. London: The MIT Press, 2018.

[9] R. Hémez, "Global Data Shock: Strategic Ambiguity, Deception, and Surprise in an Age of Information Overload, Robert Mandel, Palo Alto (CA), Stanford University Press, 2019, 272 pages," *Polit. étrangère*, vol. Hiver, no. 4, pp. XII–XII, Nov. 2019, doi: 10.3917/pe.194.0183l.

[10] P. Ferragina and F. Luccio, *Computational Thinking: First Algorithms, Then Code*. Switzerland: Springer International Publishing, 2018. doi: 10.1007/978-3-319-97940-3.

[11] P. J. Denning and M. Tedre, *Computational Thinking*. Massachusetts: The MIT Press, 2019.

[12] P. Hubwieser, "Analysis of Learning Objectives in Object Oriented Programming," in *Informatics Education –Supporting ComputationalThinking*, Roland T. Mittermeir and M. M. Sysło, Eds. Berlin, Heidelberg: Springer, 2008, pp. 142–150.

[13] F. Buitrago Flórez, R. Casallas, M. Hernández, A. Reyes, S. Restrepo, and G. Danies, "Changing a Generation's Way of Thinking: Teaching Computational Thinking Through Programming," *Rev. Educ. Res.*, vol. 87, no. 4, pp. 834–860, Aug. 2017, doi: 10.3102/0034654317710096.

[14] F. Kalelioğlu, "Characteristics of Studies Conducted on Computational Thinking: A Content Analysis," in *Computational Thinking in the STEM Disciplines*, Switzerland: Springer International Publishing, 2018, pp. 11–29. doi: 10.1007/978-3-319-93566-9_2.

[15] C. Tikva and E. Tambouris, "A systematic mapping study on teaching and learning Computational Thinking through programming in higher education," *Think. Ski. Creat.*, vol. 41, p. 100849, Sep. 2021, doi: 10.1016/j.tsc.2021.100849.

[16] G. Fessakis, V. Komis, E. Mavroudi, and S. Prantsoudi, "Exploring the Scope and the Conceptualization of Computational Thinking at the K-12 Classroom Level Curriculum," in *Computational Thinking in the STEM Disciplines*, Cham: Springer International Publishing, 2018, pp. 181–212. doi: 10.1007/978-3-319-93566-9_10.

[17] A. Deguchi, Y. Akashi, E. Hato, J. Ohkata, T. Nakano, and S. Warisawa, "Solving Social Issues Through Industry–Academia Collaboration," in *Society 5.0*, Singapore: Springer Singapore, 2020, pp. 85–115. doi: 10.1007/978-981-15-2989-4_5.

[18] B. Salgues, *Society 5.0: Industry of the future, technologies, methods and tools*. USA: John Wiley & Sons, 2018. doi: 10.1002/9781119507314.

[19] N. M. Seel, Ed., *Encyclopedia of the Sciences of Learning*, vol. 50, no. 03. Boston, MA: Springer US, 2012. doi: 10.1007/978-1-4419-1428-6.

[20] UNESCO, *Education for Sustainable Development Goals: learning objectives*. Paris, France: UNESCO, 2017. doi: 10.54675/CGBA9153.

[21] RI, *Peraturan Pemerintah Republik Indonesia Nomor 57 Tahun 2021 Tentang Standar Nasional Pendidikan*. Indonesia, 2021, p. 49.

[22] J. W. Collins and N. P. O'Brien, Eds., *The Greenwood Dictionary of Education*. London: Greenwood Press, 2003. doi: 10.5860/CHOICE.49-1829.

[23] A. D. Herlambang, A. Rachmadi, and S. H. Wijoyo, "Git and GitHub Application Training Program to Support Vocational High School Students in Collaborative Computer Programming Learning," *JPPM (Jurnal Pendidik. dan Pemberdaya. Masyarakat)*, vol. 10, no. 1, pp. 13–24, Mar. 2023, doi: 10.21831/jppm.v10i1.58550.

[24] K. Illeris, "A comprehensive understanding of human learning," in *Contemporary Theories of Learning*, London: Routledge, 2018, pp. 7–20. doi: 10.4324/9780203463321-6.

[25] E. Wenger, "A social theory of learning," in *Contemporary Theories of Learning*, New York: Routdledge,

2018, pp. 219–228. doi: 10.4324/9781315147277-16.

[26] F. Luo, M. Israel, and B. Gane, "Elementary Computational Thinking Instruction and Assessment: A Learning Trajectory Perspective," *ACM Trans. Comput. Educ.*, vol. 22, no. 2, pp. 1–26, Jun. 2022, doi: 10.1145/3494579.

[27] E. Kutay and D. Oner, "Coding with Minecraft: The Development of Middle School Students' Computational Thinking," *ACM Trans. Comput. Educ.*, vol. 22, no. 2, pp. 1–19, Jun. 2022, doi: 10.1145/3471573.

[28] H. S. Kim, S. Kim, W. Na, and W. J. Lee, "Extending Computational Thinking into Information and Communication Technology Literacy Measurement," *ACM Trans. Comput. Educ.*, vol. 21, no. 1, pp. 1–25, Mar. 2021, doi: 10.1145/3427596.

[29] R. P. Y. Lai, "Beyond Programming: A Computer-Based Assessment of Computational Thinking Competency," *ACM Trans. Comput. Educ.*, vol. 22, no. 2, pp. 1–27, Jun. 2022, doi: 10.1145/3486598.

[30] J. Corbin and A. Strauss, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. California, 2014.

[31] J. A. Rodríguez-Martínez, J. A. González-Calero, and J. M. Sáez-López, "Computational thinking and mathematics using Scratch: an experiment with sixth-grade students," *Interact. Learn. Environ.*, vol. 28, no. 3, pp. 316–327, Apr. 2020, doi: 10.1080/10494820.2019.1612448.

[32] W. A. Hazaymeh and M. K. Alomery, "The Effectiveness of Visual Mind Mapping Strategy for Improving English Language Learners' Critical Thinking Skills and Reading Ability," *Eur. J. Educ. Res.*, vol. 11, no. 1, pp. 141–150, Jan. 2021, doi: 10.12973/eu-jer.11.1.141.

[33] A. D. Herlambang, O. D. Fransisca, and T. Afirianto, "The Flipped-Classroom Instructional Procedure Development and Its Implementation Effectiveness in Improving Procedural Knowledge Learning Outcomes at Vocational High Schools," *Elinvo (Electronics, Informatics, Vocat. Educ.*, vol. 8, no. 2, pp. 201–213, Jan. 2023, doi: 10.21831/elinvo.v8i2.57845.

[34] A. D. Herlambang, R. Ririn, and A. Rachmadi, "The flipped-classroom effect on vocational high school students' learning outcomes," *Int. J. Eval. Res. Educ.*, vol. 13, no. 3, p. 1807, Jun. 2024, doi: 10.11591/ijere.v13i3.26757.

[35] C. Angeli and N. Valanides, "Developing young children's computational thinking with educational robotics: An interaction effect between gender and scaffolding strategy," *Comput. Human Behav.*, vol. 105, p. 105954, Apr. 2020, doi: 10.1016/j.chb.2019.03.018.

[36] J. Jovanović, D. Gašević, S. Dawson, A. Pardo, and N. Mirriahi, "Learning analytics to unveil learning strategies in a flipped classroom," *Internet High. Educ.*, vol. 33, pp. 74–85, Apr. 2017, doi: 10.1016/j.iheduc.2017.02.001.

[37] S. K. W. Chu *et al.*, "The effectiveness of wikis for project-based learning in different disciplines in higher education," *Internet High. Educ.*, vol. 33, pp. 49–60, Apr. 2017, doi: 10.1016/j.iheduc.2017.01.005.

[38] A. D. Herlambang, "Social Media Platforms Utilization Influence on The Vocational High School Students' Learning Interests and Learning Outcomes in Computer Network Subjects," in *2023 Eighth International Conference on Informatics and Computing (ICIC)*, Dec. 2023, pp. 1–7. doi: 10.1109/ICIC60109.2023.10381951.

[39] K. Tomita, "Does the Visual Appeal of Instructional Media Affect Learners' Motivation Toward Learning?," *TechTrends*, vol. 62, no. 1, pp. 103–112, Jan. 2018, doi: 10.1007/s11528-017-0213-1.

[40] M. Salehudin, M. Nasir, S. H. Hamzah, R. Toba, N. Hayati, and I. Safiah, "The Users' Experiences in Processing Visual Media for Creative and Online Learning Using Instagram," *Eur. J. Educ. Res.*, vol. volume-10-, no. volume-10-issue-4-october-2021, pp. 1669–1682, Oct. 2021, doi: 10.12973/eu-jer.10.4.1669.

[41] A. D. Herlambang and A. Rachmadi, "The Online Learning Interest and Learning Outcomes Through Mobile and Desktop Application Based on the Indonesian Information Technology Majoring Vocational High School Student's Perspective," in *Proceedings of the 8th International Conference on Sustainable Information Engineering and Technology*, Oct. 2023, pp. 354–360. doi: 10.1145/3626641.3627020.

[42] T. Chamidy, İ. N. S. Degeng, and S. Ulfa, "The Effect of Problem Based Learning and Tacit Knowledge on Problem-Solving Skills of Students in Computer Network Practice Course," *J. Educ. Gift. Young Sci.*, vol. 8, no. 2, pp. 691–700, Jun. 2020, doi: 10.17478/jegys.650400.

[43] H. Bai, X. Wang, and L. Zhao, "Effects of the Problem-Oriented Learning Model on Middle School Students' Computational Thinking Skills in a Python Course," *Front. Psychol.*, vol. 12, Dec. 2021, doi: 10.3389/fpsyg.2021.771221.

[44] M. Robherta, A. Dwi Herlambang, and S. Hadi Wijoyo, "The Differences of Student's Learning Outcomes and Instructional Interactions between Project Based Learning and Problem Based Learning Methods by Using Web Based Learning Technique in the Course of Videography at SMKN 1 Purwosari," *J. Inf. Technol. Comput. Sci.*, vol. 6, no. 3, pp. 225–235, Dec. 2021, doi: 10.25126/jitecs.202163305.

[45] A. Treffers, "Direct instruction and problem-solving: Critical examination of Cognitive Load Theory from the perspective of mathematics education," *Math. Enthus.*, vol. 16, no. 1–3, pp. 607–620, Feb. 2019, doi: 10.54870/1551-3440.1475.

[46] L. A. Zampetakis, L. Tsironis, and V. Moustakis, "Creativity development in engineering education: the case of mind mapping," *J. Manag. Dev.*, vol. 26, no. 4, pp. 370–380, Apr. 2007, doi: 10.1108/02621710710740110.

[47] C. Lang, A. Craig, and G. Casey, "A pedagogy for outreach activities in ICT: Promoting peer to peer learning, creativity and experimentation," *Br. J. Educ. Technol.*, vol. 48, no. 6, pp. 1491–1501, Nov. 2017, doi: 10.1111/bjet.12501.

[48] C. Zhou, A. Kolmos, and J. D. Nielsen, "A Problem and Project-Based Learning (PBL) Approach to Motivate Group Creativity in Engineering Education," *Int. J. Eng. Educ.*, vol. 28, no. 1, pp. 3–16, 2012.