Reflecting on Computational Thinking Studies for High School Education

Debby Erce Sondakh
Fakulti Pendidikan, Universiti Kebangsaan Malaysia
e-mail: debby.sondakh@unklab.ac.id

Abstrak

Berpikir komputasional telah diakui sebagai suatu kebutuhan dalam menyelesaikan masalah yang kompleks. Beberapa penelitian telah dilakukan untuk memperkenalkan keterampilan ini ke semua tingkat pendidikan. Penelitian ini bertujuan untuk meninjau penelitian tentang berpikir komputasi pada tingkat sekolah menengah. Khususnya, penelitian ini mengkaji domain penelitian, mengidentifikasi metode-metode untuk memperkenalkan berpikir komputasional, serta konsep-konsep berpikir komputasional yang diajarkan kepada pelajar. Tinjauan literatur sistematik dilakukan untuk mencapai tujuan tersebut. Hasil penelitian menunjukkan: penelitian berpikir komputasional mencakup kajian teori, pengembangan kurikulum, pengukuran, dan pengembangan alat. Kajian teori ditujukan untuk memformulasikan konsep. Selain keterampilan teknis, soft-skills telah dinyatakan sebagai elemen berpikir komputasional. Namun, perhatian untuk melibatkan soft-skills dalam penelitian masih kurang. Sebagian besar penelitian difokuskan pada integrasi berpikir komptasional ke dalam kurikulum. Coding menjadi metode yang paling banyak digunakan untuk mengajarkan berpikir komputasional. Sehingga, algorithmic thinking dan abstraction muncul sebagai keterampilan yang paling sering diajarkan atau diukur. Akhirnya, penelitian ini menggarisbawahi adanya kesenjangan untuk dikaji lebih lanjut yaitu berkaitan dengan pengukuran keterampilan berpikir komputasional dan untuk menyertakan soft-skills pada penelitian berpikir komputasional.

Kata Kunci—Berpikir komputasional, Sekolah menengah, Penyelesaian masalah

Abstract

Computational thinking (CT) has been acknowledged as a necessity in solving complex problems. Several investigations have been conducted to introduce this skill to a wide range of education level. This study intends to review the current state of CT studies in high school. In particular, it aims to examine the domain of interest of CT studies for high school, identify the different ways of promoting CT as well as the skills incorporating to the students. A systematic literature review was performed to achieve the goals. The results showed: CT studies involved theoretical study, curriculum/course, assessment, and tools development. From the theoretical point of view, several studies were pointed to formulate the skills. Other than technical skills or knowledge, soft-skills has been declared as the element of CT. However, the interest to involve soft-skills in the studies is lacking. Most of the studies still focus on integrating CT into the curriculum. Coding comes up as the most widely used method to teach CT. Subsequently, algorithmic thinking and abstraction emerge as the mostly taught or assessed skills. Finally, this study highlights the gap for further research which is CT skills assessment and to entangle the soft-skills in further CT studies.

Keywords—Computational thinking, High school, Problem-solving

Copyright @2018 | Universitas Klabat | CORIS Member | ISSN: 2541-2221 | E-ISSN: 2477-8079

1. INTRODUCTION

Over a decade ago Wing introduced a new problem-solving approach called the ‘computational thinking.' Wing's proposal is for everyone to learn and use computer scientists problem-solving techniques in solving the problem of the general domain. The idea that computer science concept would be applicable to problem-solving in other fields were sounded five decades ago. As noted by [1], in 1945, George Polya wrote about mental disciplines and methods to solve mathematics problems. Later on, Alan Perlis brought up the thought that computers would automate and transform processes in all fields. Perlis, in 1962, sounded his insight that programming would lead to the understanding of a variety of topics. Then, in 1980, Seymour Papert found that thinking like a computer, which he called ‘procedural thinking,' was a valuable component of thinking skills. He applied programming symbols and representation to teach children how to solve mathematical problems [2]. One Noble Physics scientist, L.K. Wilson, used the term ‘computational science' to refer to computation as a method to exploit existing knowledge to discovering new knowledge [3]. It can be concluded that researchers have recognized the potential of computer science towards other disciplines. However, the momentous changes took place on this topic since 2006 when Wing published her noteworthy article on Computational Thinking. CT became a ‘popular’ research topic in computer science, education, and many other disciplines.

Later, researchers put effort into reviewing the definition of CT to revisit and provide a more precise understanding of CT and to capture its essence as well. CT is defined by [1] as ways of thinking about computing. CT is the mental orientation to convert the problem into solutions [4]. In other words, CT is a problem-solving process [5]. It draws on the concepts and methodologies of computing to address problems in a broad range of subjects [6]. In 2011, Wing revised her initial definition of CT as the thought processes involved in formulating problems and their solutions that can be effectively carried out by an information-processing agent; a human or machine or combinations of humans and machines [7], which later concluded by [8] as the thought processes involved in formulating problems into computational solutions, i.e., the algorithms. Thinking like a computer scientist is the essence of CT [9].

As noted earlier, CT has drawn researchers attention, particularly since it has acknowledged as critical skills required in the Fourth Industrial Revolution (4th IR) workplaces. Recent studies have shown that efforts to equip the young generations with CT skills continue to be carried out, investigating both teaching and assessing issues. In this study, the researcher provides a flashback of what has been done in CT, in particular for high school level. Evaluating the current state of the CT studies is beneficial for educators to have insight on CT development as a general, moreover, in particular, the CT technology and content necessary for high school level pupils, as well as strategies and methods which reliable to be used as best practices to teach and assess CT skills. Additionally, this study reveals the gaps in CT studies that further can be improved.

2. RELATED WORKS

A literature review presented by [10] investigated the development of CT through programming. From the review, authors propose to conduct more intervention studies highlighting the computational practices and computational perspectives in the regular classroom. They further recommend employing constructionism-based problem-solving learning which entails information processing, scaffolding, and reflection activities to foster the computational practices and computational perspectives.

One study by [2] explored the issues of CT in higher education level. The authors were particularly interested in the issue concerning the application of CT skills outside of STEM fields. From the review, authors concluded that the difference between K-12 and research-oriented domain of higher education resulting in a different approach to incorporate CT in these

two environments. Overall, much of the CT work has focused on definitional issues, incorporation of computational thinking in computer science curricula and the infusion of CT strategies into non-computer science disciplines. The game-based learning was found the potential for teaching CT within computer science and the STEM disciplines in higher education. Application of CT is shaped by the different perception of CT terminology and the issues of the non-computable problem, in particular for the fields outside of computer science and non-STEM.

A systematic mapping study carried out by [11] focused on the approaches used to observe and assess the development of CT skills. The mapping results have shown that the vast majority of the studies considered the general problem-solving. Direct programming course came up as the mostly used pedagogical approach using visual programming as the tools to teach CT. The most common skills assessed comprising solving problem, developing algorithms, and applying abstraction. Additionally, researchers have mostly used coding and multi-choice questionnaire for assessing CT. Therefore, concepts such as abstraction, algorithm, decomposition, generalization, and related process are associated with CT. Studies in CT obtained in recent academic literature have shown wide-range of skills.

3. RESEARCH METHODOLOGY

A huge number of studies have been conducted in CT fields, and one may find different findings. Consequently, seizing a clear overall picture that defines the core idea might be problematic. On the other hand, given the fact of CT importance in 4th IR, one need to have an understanding of well-defined CT concept. Therefore, this study was conducted to uncover the accomplishment in CT studies for high school level, in a decade since CT is sounded as vital skills digital citizen needs to master. For doing so, the following objectives are defined to direct this study, which is to bring forth: The domain of interest of CT studies in high school, the approaches for promoting CT, the CT skills incorporated in high school, the approaches to assessing CT skills, and the tools used for promoting CT.

3.1. Data Collection

Subsequent to the definition of the objectives is the process of collecting the publications related to CT for high school (upper secondary) education. The term **"Computational Thinking AND (High School OR Secondary Education"** came up as the search keywords. Three electronic databases were chosen, they are IEEE, ACM, and SpringerLink. Researcher focused on research articles published in the range of 2007 to 2017.

Table 1. Data Collected

Databases	Search Results
ACM	190
IEEE	518
SpringerLink	43
Science Direct	42
TOTAL	793

3.2. Papers Screening

The next process to do screening on the articles resulting from data collection (see Figure 1). It starts screening based on some exclusion criteria described below:

a) Review article, abstract only, short paper, the chapter of a book, and duplicated articles.
b) No access to the full paper.
c) Paper published other than in English

d) Secondary studies

As many as 164 articles were removed after the initial screening. Then, the title screening was conducted. The articles included are the ones containing one of the following the terms 'Computational thinking', 'Computer Science,' 'Computer Science Education,' 'Computing,' 'Programming,' and other computer science related terms. As a result, 189 articles were excluded from the pool. Following the title screening was articles selection based on their abstracts, which removed 190 articles. Lastly, only those that describe the studies in the education context, particularly for high school, are chosen; 184 articles were excluded. Conclusively, the remaining accepted articles from the extraction process are as many as 66 articles.

4. FINDING AND DISCUSSION

This section elaborates the results of this study. The graph in Figure 1 shows the publication trends of computational thinking. As noted earlier, the time delimit for this study is in the range of 2007 to 2017. However, the first chosen study is from 2009. In three consecutive year since 2009, there was only one study found in each year. The increase in the number of CT articles point towards high school level starting in 2013: 4, 10, 13, 14, and 19 studies respectively in 2013, 2014, 2015, 2016, and 2017.

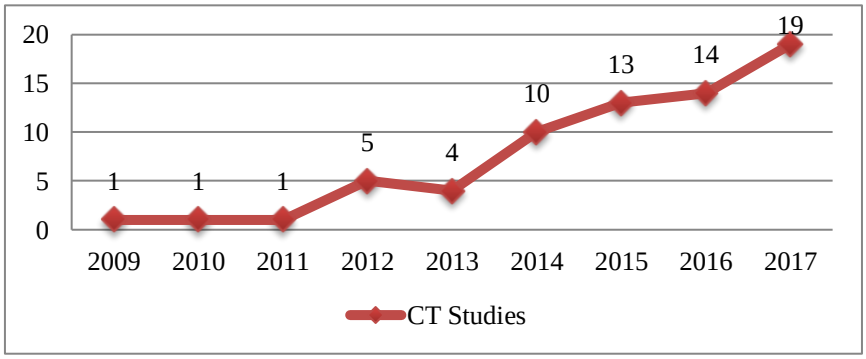

Figure 1. Computational Thinking Publication Trend

4.1. Domain of Interest

This study classifies five domain of interest of CT studies, namely theory, curriculum/course, assessment, tools development, and informal program. The term theory refers to the definition of CT concepts, in particular, the skills that describe this computer science-based problem solving, as well as the approaches to teaching CT. The term curriculum/course includes curriculum development, module, and courses offer in regular classrooms. The informal program, otherwise, is the after-school programs for teaching CT. It covers the workshop, boot camps, and outreach activities. Assessment includes studies that intended for measuring CT skills. Tools development refers to the tools created to teach and assess CT skills.

Of the 66 studies identified in this review as the ones targeting high school students, most of the studies focused on distributing CT skill to the students. The graph in Figure 2 shows CT studies include developing curriculum/course (30 studies) and informal program (15 studies) as well. Following curriculum/course and informal program are tools development (15 studies), assessment (6 studies) and theory (3 studies).

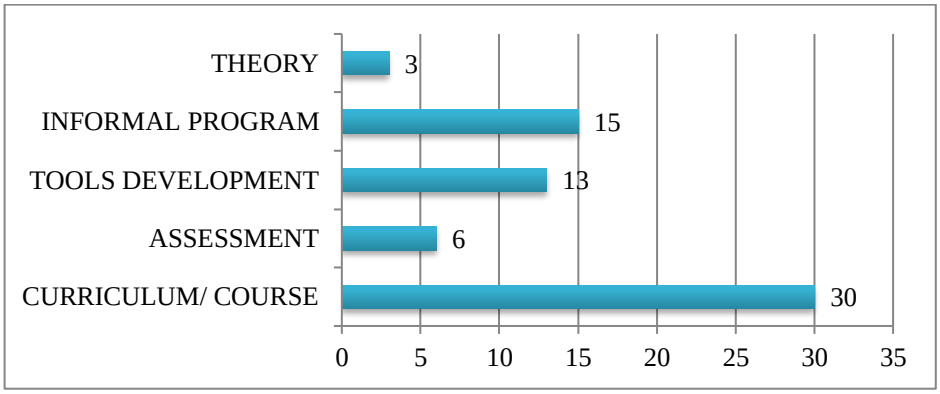

Figure 2. Approaches to Promote Computational Thinking Skills

4.2. Definition of CT for High School

Barr & Stephenson [5] reported a join-hand project, in 2009, by the Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE). The project aimed at developing an operational definition of CT for K-12. They worked out to formulate the core concept of CT for K-12, which are the abstraction, automation, algorithm, data collection and analysis, programming, test and debug, simulation, innovation and creativity, group problem-solving, and learning strategies. In addition to the aforementioned capabilities, the findings also emphasized on the disposition (attitudes) applicable to CT, including: confidence in dealing with complexity, persistence in working with difficult problems, ability to handle ambiguity, ability to deal with open-ended problems, work with others, and the awareness of one's strengths and weaknesses.

A recent study by [12] recommended a new definition of CT which resulting from the composite of CT and Agile philosophy of Software Engineering domain, called ‘Cooperative Thinking' (CooT). CooT is then defined as "the ability to describe, recognize, decompose problems and computationally solve them in teams in a socially sustainable way." Agile's key factor is a team-related skill. Therefore, by merging CT and this Agile principle may enhance the current efforts to establish CT as a fundamental skill. Moreover, the authors also argued that soft-skills would rise in importance in the future. Adding teamwork and soft-skill would be beneficial in solving problems. The authors argued that Agile principles might enhance the current efforts to establish CT.

One study by [13] suggested a model-based thinking approach to teaching CT. It comprises the mental model and computerized model. The mental model is required to analyze and understand phenomena as well as design and construct artifacts, while the computerized model that embeds in computer-based system role as a tool for creating models of phenomena and concepts from the real or imaginary world. However, the authors provided no further reflection on how to practically apply the proposed approach to teaching CT.

4.3. CT Educational Approaches

There is a broad-based discussion around the educational approaches to integrating CT into the high school curricula. As shown in Table 2, to teach CT, 15 studies employed coding (programming), 12 studies interested in developing teaching module, three studies employed unplugged activities, game-based, and framework development. Programming is widely used to teaching CT, including Scratch [14]–[21], followed by Phyton [14], [17], [18], [20], App Inventor [15], [21] Alice [22], Microsoft Kodu [22], Lego Mindstorms [22], Logo [17], C/C++ and BYOB [15]. Android programming was used in robot development [23], [24]. Other than using the available tools, some studies attempted to create new programming-based teaching tools [25]–[30], and game-based tools [31]–[33].

Table 2. Approaches to Teach CT in Regular Classroom

Approaches	References	Frequency
Coding	[22], [14], [17], [15], [16], [34], [18], [19], [20], [35], [36], [23], [37], [21], [24]	15
Module	[38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49]	12
Unplugged	[22], [15] [17], [50]	4
Game-based	[51]	1
Framework	[52]	1

In addition to teaching CT in regular classrooms, researchers have introduced after-school activities, which called 'informal program' in this study, as an alternative to disseminate CT skills. Recently, the publication trend of studies related to infusing CT through the informal program has increased. As Figure 3 shown, the informal program includes summer camp (6 studies), workshop (6 studies), and outreach program (3 studies) that teach CT using coding (10 studies), game-based (3 studies), unplugged approaches (3 studies), classroom module (1 study) (see Table 3).

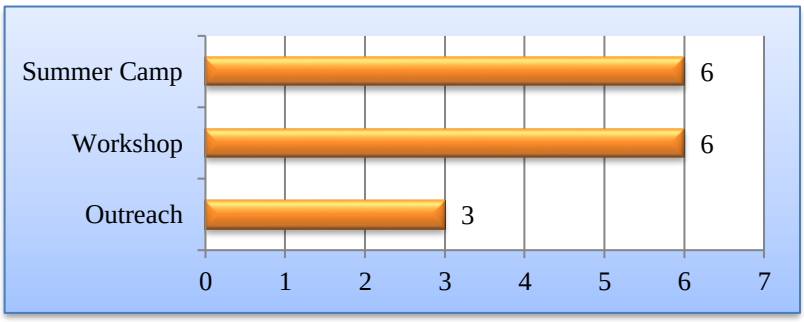

Figure 3. After-school Program

Table 3. Approaches to Teach CT in Informal Program

Approaches	References	Frequency
Coding	[53], [54], [55], [56], [57], [58], [59], [60], [61], [62]	10
Game-based	[63], [64], [65],	3
Unplugged	[66], [54], [57]	3
Module	[67]	1

As noted earlier, many studies emphasize coding as the commonly used method for encouraging CT. Nevertheless, no further enlightenment on the problem-solving scenario employed in the coding tasks. Among the examined studies, particularly the ones that concentrate on coding as a method to train CT skills, the motivation was to teach programming to novice users. Such practices may lead impression that CT is indistinguishable from coding. It is contrary to the conception that CT is not computer programming; on the other hand, thinking like a computer scientist that implies more than being able to do programming [68]. Moreover, emphasizing on programming language can distract and lead students to fail in concentrating on what the 'brainy' task is [69]. CT is not necessarily engaging computer programming; otherwise, it should point out the thinking process rather than a product [70]. In order to address this situation, computer science unplugged is introduced as another road to acquiring CT skill. Using unplugged method enables the students to nurture CT skills without the computer. This approach to teaching computing concepts using constructivist activities with the absent of computer, which was established by Tim Bell, has been well-acknowledged both in the regular classroom (see Table 2) and after-school program (see Table 3).

4.4. CT Skills for High School Students

The skills promoted to high school students as identified in this study are presented in Table 4, row ‘a' for the regular classroom while row ‘b' for the informal program. The ones without ‘b’ field stated the absent CT studies that examined such skill. The skills are varying, and each study employed a different set of skills. However, the algorithmic thinking was found as the most emphasized CT skill, both in the regular classroom and informal program. It can be so because following the findings that coding has been the most popular approach to teach CT. Following the algorithmic thinking are the abstraction, data, modeling, simulation, logical thinking, decomposition, debugging, automation, and generalization. This result highlights the general programming skills.

It is interesting to note [68] declared that CT represents a set of applicable attitude and skill everyone would use in solving problems, designing system, and understanding human behavior. In other words, the attitude or soft-skill should be included in CT concepts. However, the result showed, the awareness of merging attitude in CT is still low. Only very few studies that took it into account [21], [40], [54], [55], [64]. Problem-solving, teamwork, communication, collaboration, and persistence were the attitudes found in the examined studies. Computing education should not focus only, on technical skills, but as well encourage soft-skills development as a critical element of problem-solving ability. As [12] point out, educators should not only promote coding skills and provide knowledge, but also collaboration and teamwork skills to deal with difficult problems which are too hard to be solved individually.

Table 4. Approaches to Teach CT in Informal Program

Content/Skills		References	Frequency
Algorithmic thinking	a	[52], [38], [39], [41], [22], [14], [15], [42], [18], [44], [17], [45], [19], [20], [36], [46], [23], [37], [50], [47], [21], [48], [49], [51], [24]	25
	b	[53], [57], [58], [59], [60], [61], [64], [55], [56], [66], [67]	11
Abstraction	a	[71], [22], [15], [42], [18], [44], [17], [19], [34], [46], [49], [51], [24]	13
	b	[57], [56], [66]	3
Logical thinking	a	[41], [43], [51]	3
	b	[62], [64], [55], [56], [66]	5
Modeling	a	[52], [14], [18], [44], [45], [46], [49]	7
	b	[61], [65], [66]	4
Debugging	a	[46], [51]	2
	b	[57], [62], [61]	3
Decomposition	a	[42], [17], [46], [49], [51], [24]	6
	b	[62], [56]	2
Generalization	a	[17], [51], [24]	3
Data	a	[52], [41], [14], [15], [44], [45], [34], [46], [37], [51]	10
	b	[53], [61]	2
Design	a	[52]	1
Automation	a	[44], [20], [36], [46]	4
	b	[64]	1
Parallelization	a	[51]	1
Simulation	a	[39], [14], [18], [44], [45] [46], [49], [51]	8
	b	[62]	1
Evaluation	a	[40], [18] [17]	3
	b	[66]	1
Pattern Recognition	a	[42], [24]	2

Proof	b	[66]	1
	b	[67]	1
Graph	a	[39], [40], [38]	3
Coordination	a	[52]	1
IR	a	[40]	1
Networking	a	[52]	1
	b	[67]	1
Hardware	a	[48]	1
	b	[67]	1
Modularization	b	[57]	1
HCI	a	[34]	1
Multimedia	a	[19]	1
Creativity	a	[15], [36], [21]	3
Problem-solving	a	[21]	1
	b	[55]	1
	b	[55], [64]	2
Communication	a	[40]	1
	b	[54]	1
Collaboration	a	[21]	1
Persistence	a	[21]	1

4.5. CT Assessment

Assessment is also a concern in CT skills development. According to Román-González [72], assessment supports the process of merging CT into the curriculum. It is required to measure students' understanding of CT concepts. To measure CT abilities, researchers implemented different types of approaches, as shown in Figure 4. The commonly used CT assessment in high school level is coding (2 studies), multiple choice (2 studies), survey questionnaire (2 studies), and task-based (1 study). The skills assessed is presented in Table 5.

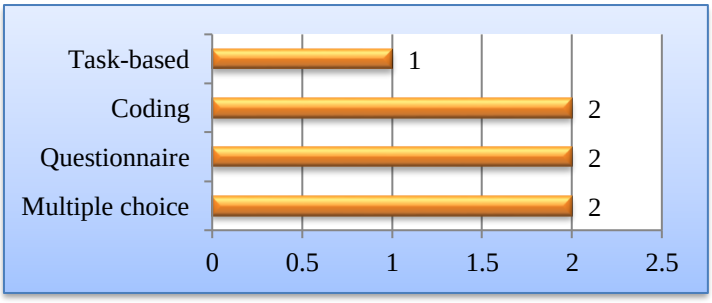

Figure 4. Approaches to Assess Computational Thinking

In [73], the effect of abstraction thinking level on ones programming understanding was examined. The authors adopted Behavioral Identification Form to measure abstract thinking. A study by [74] proposes a framework for measuring CT in block-based programming environment Alice. The framework was applied to analyze a dataset of performance assessment task from 'Fairy Assessment in Alice,' a programming-based assessment developed by [75], to gain an understanding of students' programming process.

The work in [76] combines two approaches to evaluate students' CT skills. First, the coding approach was applied in a small group whereby the students solve robot programming problem guided by some worksheets. Students' CT skills were evaluated during the coding activity. Second, a questionnaire was administered to asked students to solve programming problems and practice CT skills during their solution process, after attending several training

seminars. Therefore, the purpose of the questionnaire was indeed to assess students' level of CT skills development.

Wang, Huang & Hwang [16] proposed an integrated Scratch and project-based learning to promote CT, as well as a pre-test and post-test aimed at testing the effectiveness of the proposed learning approach. The pre-test was conducted to evaluate students' prior knowledge before the learning activity, while the post-test for evaluating students' learning achievements. Both tests comprise 25 of multi-choice items and multi-select items. Experts in computer science, with a minimum of ten years of experience, developed the test. However, the authors do not reveal how the instrument was developed, and whether it is validated.

Table 5. Computational Thinking Skills Assessed

Skills	References	Frequency
Algorithmic Thinking	[76], [78], [74], [77]	4
Abstraction	[73], [76], [78], [74]	4
Decomposition	[76], [78], [74]	3
Generalization	[76]	1
Data	[78]	1
Debugging	[74]	1
Iterative refinement	[74]	1
Modularization	[76]	1
IT Literacy	[78]	1
HCI	[77]	1
Web Design	[77]	1

Snow et al. [77], created a validated instrument that measures the practices of CT. The authors argued that relying on the programming constructs to infer students' knowledge of computer science is not enough. The reason why is the students may use code deprived of understanding what it does. Moreover, it only assesses the conceptual knowledge. Therefore, the authors build the 'Exploring Computer Science,' an instrument comprises a set of scenario represent CT-based problem-solving context. Each scenario has a related set of questions (items) to measure CT practices.

5. CONCLUSION

The CT studies for high school level have considered several issues, including theoretical, pedagogical, and assessment. Additionally, develop tools for teaching and assessment is another concern in CT studies. However, much of the studies found in this review have focused on the efforts to incorporate CT into the classroom. Coding comes up as the common way to teach CT to high school students, both in the regular classroom and during after-school activities. Coding has also been the leading approach in CT assessment. Furthermore, visual programming transpires as the well-known tools employed in CT studies. Indisputably, this programming environment has made computing education more accessible to a broad range of students, even to those who are a novice. Nevertheless, it is not always clear what problem-solving concepts the students have actually learned through the use of the coding approach. In addition to coding, the unplugged approach has been introduced to promote CT away from a computer. Subsequent to coding, algorithmic thinking was found as leading CT skills taught to the students. Following the algorithmic thinking are the abstraction, data, modeling, simulation, logical thinking, decomposition, debugging, automation, and generalization. Other than this technical knowledge of CT, some studies have acknowledged soft-skills as other essential skills required in CT. Problem-solving, teamwork, communication, collaboration, and persistence were the attitudes found in the examined studies.

Copyright @2018 | Universitas Klabat | CORIS Member | ISSN: 2541-2221 | E-ISSN: 2477-8079

6. SUGGESTION

There still much remains to be done concerning the introduction of CT skill to secondary education pupils.

- Establish studies that consider more on CT assessment. CT might be in its infancy stage; however, assessment is required as a crucial supporting component in promoting CT to the students, and to examine the outcomes of teaching CT.
- Build a CT framework that incorporates soft-skills. By considering the necessity of soft-skills in the future workforces, bringing them together in CT teaching and assessment is a must to equip the students with relevant required skills.

REFERENCES

[1] Guzdial, M., 2008, Education Paving the way for computational thinking, *Commun. ACM*, vol. 51, p. 25.

[2] Czerkawski, B. and Lyman III, E., 2015, Exploring issues about computational thinking in higher education, *TechTrends*, vol. 59, pp. 57–65.

[3] Denning, P.J., 2007, Computing is a natural science, *Commun. ACM*, vol. 50, p. 13.

[4] Denning, P.J., 2009, The profession of IT Beyond computational thinking, *Commun. ACM*, vol. 52, p. 28.

[5] Barr, B.V. and Stephenson, C., 2011, Computational Thinking to K-12 : What is Involved and What is the Role of the Computer Science Education Community ?, vol. 2, p. 48–54.

[6] Qin, H., 2009, Teaching computational thinking through bioinformatics to biology students, *ACM SIGCSE Bull.*, vol. 41, p. 188–191.

[7] Wing, J.M., 2011, Research notebook: Computational thinking—What and why?, *The Link Magazine*, p. June 23, Retrieved on March 2017 from https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why.

[8] Aho, A.V., 2012, Computation and computational thinking, *Comput. J.*, vol. 55, p. 833–835.

[9] Grover, S. and Pea, R., 2013, Computational Thinking in K-12: A Review of the State of the Field, *Educ. Res.*, vol. 42, p. 38–43.

[10] Lye, S.Y. and Koh, J.H.L., 2014, Review on teaching and learning of computational thinking through programming: What is next for K-12?, *Comput. Human Behav.*, vol. 41, p. 51–61.

[11] de Araujo, A.L.S.O., Andrade, W.L., and Serey Guerrero, D.D., 2016, A systematic mapping study on assessing computational thinking abilities, *Proceeding of 2016 IEEE Frontiers in Education Conference*, Erie, USA, Oct 12-15.

[12] Missiroli, M., Russo, D. and Ciancarini, P., 2017, Cooperative Thinking, or: Computational Thinking Meets Agile," *Proceeding of 30th IEEE Conference on Software Engineering Education and Training*, Savannah, USA, Nov. 7-9.

[13] Nowack, P. and Caspersen, M.E., 2014, Model-Based Thinking and Practice A Top-down Approach to Computational Thinking, *Proceeding of Koli Calling*, Koli, Finland, Nov. 20-23.

[14] Arraki, K. *et al.*, 2014, DISSECT: An experiment in infusing computational thinking in K-12 science curricula, *Proceedings of Frontiers in Education Conference*, Madrid, Spain, Oct 22-25.

[15] Giordano, D. and Maiorana, F., 2014, Use of cutting edge educational tools for an initial programming course, *Proceeding of IEEE Global Engineering Education Conference*, Istanbul, Turkey, April 3-5.

[16] Wang, H.Y., Huang, I. and Hwang, G.J., 2014, Effects of an integrated scratch and project-based learning approach on the learning achievements of gifted students in computer courses, *Proceeding of 3rd Int. Conf. Adv. Appl. Informatics*, Kitakyushu,

Japan, Aug 3-Sept 4.
[17] Dorling, M. and White, D., 2015, Scratch: A Way to Logo and Python, *Proceeding of 46th ACM Tech. Symp. Comput. Sci. Educ.*, Kansas City, USA, March4-7.
[18] Burgett, T., Folk, R., Fulton, J., Peel, A., Pontelli, E. and Szczepanski, V., 2015, DISSECT: Analysis of pedagogical techniques to integrate computational thinking into K-12 curricula, *Proceeding of Frontiers in Education Conference,* El Paso, USA, Oct 21-24.
[19] Basogain, X., Olabe, M.A., Olabe, J.C., Ramirez, R., Del Rosario, M. and Garcia, J., 2016, PC-01: Introduction to computational thinking: Educational technology in primary and secondary education, *Proceeding of Int. Symp. Comput. Educ.*, Salamanca, Spain, Sept. 13-15.
[20] Chang, C., Chin, Y.L., and Chang, C.K., 2016, Experimental Functionality Development for Scratch Mathematical and Statistics Extensions, *Proceeding of International Computer Symposium*, Chiayi, Taiwan, Dec. 16-17.
[21] da S. Eloy, A.A., Martins, A.R.Q., Pazinato, A.M., Lukjanenko, M.F.S.P., and de Lopes, D.R., 2017, Programming Literacy: Computational Thinking in Brazilian Public Schools, *Proceeding of Conference on Interaction Design and Children*, Stanford, USA, June 27-30.
[22] Touretzky, D.S., Marghitu, D., Ludi, S., Bernstein, D. and Ni, L., 2013, Accelerating K-12 computational thinking using scaffolding, staging, and abstraction, *Proceeding of 44th ACM Tech. Symp. Comput. Sci. Educ.*, Denver, Colorado, March 6-9.
[23] Ball, D., Tofel-Grehl, C., and Searle, K.A., 2017, Sustaining Making in the Era of Accountability, *Proceeding of 7th Annu. Conf. Creat. Fabr. Educ.*, Stanford, USA, Oct. 21-22.
[24] Martín-Ramos, P., *et al.*, 2017, First exposure to Arduino through peer-coaching: Impact on students' attitudes towards programming, *Comput. Human Behav.*, vol. 76, pp. 51–58.
[25] Cross, J., Bartley, C., Hamner, E. and Nourbakhsh, I., 2013, A visual robot-programming environment for multidisciplinary education, *Proceeding of IEEE Int. Conf. Robot. Autom.*, Karlsruhe, Germany, May 6-10.
[26] Gorman, J., Gsell, S. and Mayfield, C., 2014, Learning relational algebra by snapping blocks, *Proceeding of 45th ACM Tech. Symp. Comput. Sci. Educ.*, Atlanta, USA, March 5-8.
[27] Caglar, F. *et al.*, 2015, Cloud-hosted simulation-as-a-service for high school STEM education, *Simul. Model. Pract. Theory*, vol. 58, 2015, p. 255–273.
[28] de Lima, J.P.C., Carlos, L.M., Schardosim Simão, J.P., Pereira, J., Mafra, P.M. and da Silva, J.B., 2016, Design and implementation of a remote lab for teaching programming and robotics, *IFAC-PapersOnLine*, vol. 49, p. 86–91.
[29] Gonzalez-Sacristan, C., Garcia-Saura, C., and Molins-Ruano, P., 2016, Phogo: A Low Cost, Engaging and Modern Proposal to Learn How to Program, *Proceeding of Fourth Int. Conf. Technol. Ecosyst. Enhancing Multicult.*, Salamanca, Spain, Nov. 2-4.
[30] Goyal, S., Vijay, R.S., Charu, M. and Pratul, K., 2016, Code Bits: An Inexpensive Tangible Computational Thinking Toolkit For K-12 Curriculum, *Proceedings of the Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, Eindhoven, Netherlands, Feb.14-17.
[31] Ahmadi, N., Jazayeri, M., and Landoni, M., 2012, Helping novice programmers to bootstrap in the cloud: Incorporating support for computational thinking into the game design process, *Proceeding of 12th IEEE Int. Conf. Adv. Learn. Technology*, Rome, Italy, July 4-6.
[32] Bauer, A. Butler, E., and Popovic, Z., 2015, Approaches for teaching computational thinking strategies in an educational game: A position paper, *Proceeding of 2015 IEEE Blocks and Beyond Workshop,* Atlanta, USA, Oct. 22.
[33] B. Nikaido and J. Ventura, "Code puzzles - Robot Chronicle," *Proceeding of IEEE SOUTHEASTCON*, Norfolk, USA, March 30-April 3.

[34] Brady, C. et al., 2016, All roads lead to computing: Making, participatory simulations, and social computing as pathways to computer science, *IEEE Transactions on Education*, pp. 1–8.
[35] Fronza, I., El Ioini, N., and Corral, L., 2016, Teaching Software Design Engineering Across the K-12 Curriculum, *Proceeding of 17th Annu. Conf. Inf. Technol. Educ.*, Boston, USA, Sep. 28-Oct.1.
[36] Helms, M., Moore, R., Edwards, D. and Freeman, J., 2016, STEAM-based interventions: Why student engagement is only part of the story, *Proceeding of Research on Equity and Sustained Participation in Engineering, Computing, and Technology*, Atlanta, USA, Aug. 11-13.
[37] Bubno, K. and Takacs, V.L, 2017, The mathability of word problems as initial computer programming exercises, *Proceeding of 8th IEEE Int. Conf. Cogn. Infocommunications*, Debrecen, Hungary, Sept. 11-14.
[38] Gibson, J. P., 2012, Teaching graph algorithms to children of all ages, *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*, Haifa, Israel, July 3-5.
[39] Goldberg, D. S., Feld, J.A., Grunwald, D., Lewis, C. and Hug, S., 2012, Engaging Computer Science in Traditional Education : The ECSITE Project, *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*, Haifa, Israel, July 3-5.
[40] Settle, A. *et al.*, 2012, Infusing computational thinking into the middle- and high-school curriculum, *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*, Haifa, Israel, July 3-5.
[41] Yevseyeva, K. and Towhidnejad, M., 2012, Work in progress: Teaching computational thinking in middle and high school, *Proceeding of IEEE Frontiers in Education Conference*, Seattle, USA, Oct. 3-6.
[42] Towhidnejad, M., Kestler, C., Jafer, S. and Nicholas, V., 2014, Introducing Computational Thinking through Stealth Teaching, *Proceeding of IEEE Frontiers in Education Conference*, Madrid, Spain, Oct. 22-25.
[43] Brancaccio, A., Marchisio, M., Palumbo, C., Pardini, C., Patrucco, A. and Zich, R., 2015, Problem Posing and Solving: Strategic Italian Key Action to Enhance Teaching and Learning Mathematics and Informatics in the High School, *Proceedings of International Computer Software and Applications Conference*, Taichung, Taiwan, July 1-5.
[44] Chuang, H.C., Hu, C.F., Wu, C.C. and Lin, Y.T., 2015, Computational thinking curriculum for K-12 education - A Delphi survey, *Proceedings of International Conference on Learning and Teaching in Computing and Engineering*, Taipei, Taiwan, April 9-12.
[45] Nesiba, N., Pontelli, E., and Staley, T., 2015, DISSECT: Exploring the relationship between computational thinking and English literature in K-12 curricula, *Proceedings of Frontiers in Education Conference*, El Paso, USA, Oct. 21-24.
[46] Weintrop, D. *et al.*, 2016, Defining Computational Thinking for Mathematics and Science Classrooms, *Journal of Science Education and Technology*, vol. 25, p. 127–147.
[47] Cox, R., Bird, S., and Meyer, B., 2017, Teaching Computer Science in the Victorian Certificate of Education, *Proceeding of 2017 ACM SIGCSE*, Seattle, USA, March 8-11.
[48] Fields, D.A., Kafai, Y.B., Nakajima, T. and Goode, J., 2017, Teaching practices for making e-textiles in high school computing classrooms, *Proceeding of 7th Annual Conference on Creativity and Fabrication in Education*, Stanford, USA, Oct. 21-22.
[49] Gautam, A., Bortz, W.E.W. and Tatar, D., 2017, Case for Integrating Computational Thinking and Science in a Low-Resource Setting," *Proceeding of Ninth Int. Conf. Inf. Commun. Technol. Dev.* Lahore, Pakistan, Nov. 16-19.
[50] Conde, M.A., Camino, F.-L., Garcia-Penalvo, F.J,. Rodriguez-Sedano, F.J., Guerrero-Higueras, A.M. and Matellan-Olivera, V., 2017, Promoting Computational Thinking in

K-12 Students by Applying Unplugged Methods and Robotics, *Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality*, Cádiz, Spain, Oct. 18-20.

[51] Pellas, N. and Vosinakis, S., 2017, How can a simulation game support the development of computational problem-solving strategies?, *Proceeding of IEEE Glob. Eng. Educ. Conf.*, Athens, Greece, April 25-28.

[52] Starr, C.W., Bergman, D. and Zaubi, P., 2009, The development and implementation of a context-based curricular framework for computer science education in high schools," *Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education*, Paris, France, July 6-9.

[53] Chatzinikolakis, G. and Papadakis, S., 2014, Motivating K-12 students learning fundamental Computer Science concepts with App Inventor: Mobile application development in secondary education," *Proceeding of 2014 Int. Conf. Interact. Mob. Commun. Technol. Learn.*, Thessaloniki, Greece, Nov. 13-14.

[54] Pollock, L., Mouza, C., Atlas, J. and Harvey, T., 2015, Field Experiences in Teaching Computer Science : Course Organization and Reflections, *Proceeding of 46th ACM Tech. Symp. Comput. Sci. Educ.*, Kansas City, USA, March 4-7.

[55] Lakanen, A.-J. and Isomöttönen, V., 2015, What Does It Take to Do Computer Programming?," *Proceeding of 46th ACM Tech. Symp. Comput. Sci. Educ.*, Kansas City, USA, March 4-7.

[56] Lee, J., Jun, S., Kim, K. and Yoon, I., 2016, Summer programming boot camp, teach high schoolers first and let them teach middle schoolers, *Proceedings of International Conference on Computational Science and Computational Intelligence*, Las Vegas, USA, Dec. 15-17.

[57] Guenaga, M., Menchaca, I., Garaizar, P. and Eguíluz, A., 2017, Trastea.club, an initiative to develop computational thinking among young students, *Proceeding of 5th Int. Conf. Technol. Ecosyst. Enhancing Multicult.*, Cádiz, Spain, Oct. 18-20.

[58] Papavlasopoulou, S., Sharma, K., Giannakos, M. and Jaccheri, L., 2017, Using Eye-Tracking to Unveil Differences Between Kids and Teens in Coding Activities, *Proceeding of 2017 Conf. Interact. Des. Child.*, Stanford, USA, June 27-30.

[59] Eguiluz, A., Guenaga, M., Garaizar, P. and Olivares-Rodriguez, C., 2017, Exploring the progression of early programmers in a set of computational thinking challenges via clickstream analysis, *IEEE Trans. Emerg. Top. Comput.*, preprint vol., p. 1-6.

[60] Mufeti, T.K. and Sverdlik, W., 2017, Introducing computer programming in secondary schools: A case study of NAMTOSS, *Proceeding of IST-Africa Week Conference*, Windhoek, Namibia, May 30-June 2.

[61] Ziaeefard, S., Miller, M.H., Rastgaar, M and Mahmoudian, N., 2017, Co-robotics hands-on activities: A gateway to engineering design and STEM learning, *Rob. Auton. Syst.*, vol. 97, p. 40–50.

[62] Wagh, A., Gravel, B. and Tucker-Raymond, E., 2017, The role of computational thinking practices in making: How beginning youth makers encounter & appropriate ct practices in making, *Proc. 7th Annu. Conf. Creat. Fabr. Educ.*, Stanford, USA, Oct. 21-22.

[63] Lee, M.J. *et al.*, 2014, Principles of a debugging-first puzzle game for computing education, *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing*, Melbourne, Australia, July 28-Aug. 1.

[64] Roscoe, J.F., Fearn, S., and Posey, E., 2014, Teaching computational thinking by playing games and building robots, *Proceedings of 2014 International Conference on Interactive Technologies and Games*, Nottingham, UK, Oct. 16-17.

[65] Lamprecht, A.L., Margaria, T., and McInerney, C., 2016, A Summer Computing Camp Using ChainReaction and jABC, *Proceedings of International Computer Software and Applications Conference*, Atlanta, USA, June 10-14.

[66] Curzon, P., 2013, Cs4Fn and Computational Thinking Unplugged, *Proc. 8th Work. Prim.*

Second. Comput. Educ., Aarhus, Denmark, Nov. 11-13.
[67] Feaster, Y., Ali, F., Zhai, J. and Hallstrom, J.O., 2014, Serious Toys : Three Years of Teaching Computer Science Concepts in K-12 Classrooms, *Proceedings of the Conference on Innovation and Technology in Computer Science Education*, Uppsala, Sweden, June 21-25.
[68] Wing, J.M., 2006, Computational thinking, *Commun. ACM*, vol. 49, p. 33–35.
[69] Davies, S. and Washington, M., 2008, The Effects of Emphasizing Computational Thinking in an Introductory Programming Course, *Proceeding of 38th ASEE/IEEE Frontiers in Education Conference*, Saratoga Springs, NY, Oct. 22-25.
[70] Roberts, J., 2016, Computational thinking, *SET*, vol. 1, p. 3–7.
[71] Miller, C.S., Perković, L. and Settle, A., 2013, File references, trees, and computational thinking, *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, Ankara, Turkey, June 26-30.
[72] Román-gonzález, M., 2015, Computational Thinking Test : Design Guidelines and Content Validation Computational Thinking Test : Design Guidelines and Content Validation, *Proceeding of EDULEARN15 Conf.*, Barcelona, Spain, July 6-8.
[73] Park, C.J., Hyun, J. S., and Heuilan, J., 2014, Effects of gender and abstract thinking factors on adolescents' computer program learning, *Proceedings Frontiers in Education Conference*, El Paso, USA, Oct. 21-24.
[74] Grover, S., Basu, S., Bienkowski, M., Eagle, M., Diana, N. and Stamper, J. 2017, A Framework for Using Hypothesis-Driven Approaches to Support Data-Driven Learning Analytics in Measuring Computational Thinking in Block-Based Programming Environments, *ACM Trans. Comput. Educ.*, vol. 17, p. 1–25.
[75] Werner, L., Denner, J. and Campe, S., 2012, The Fairy Performance Assessment : Measuring Computational Thinking in Middle School," *Proc. 43rd ACM Tech. Symp. Comput. Sci. Educ.*, Raleigh, USA, Feb. 29-March 3.
[76] Atmatzidou, S. and Demetriadis, S., 2016, Advancing students ' computational thinking skills through educational robotics : A study on age and gender relevant differences, *Rob. Auton. Syst.*, vol. 75, p. 661–670.
[77] Snow, E., Rutstein, D., Bienkowski, M. and Xu, Y., 2017, Principled Assessment of Student Learning in High School Computer Science, *Proceeding of ACM Conf. Int. Comput. Educ. Res.*, Tacoma, USA, Aug. 18-20.
[78] Dagienė, V., Stupurienė, G. and Vinikienė, L., 2016, Promoting Inclusive Informatics Education Through the Bebras Challenge to All K-12 Students, *Proceeding of 17th Int. Conf. Comput. Syst. Technol.*, Palermo, Italy, June 23-24.