Teaching Strategies Guidelines to Foster the Computational Thinking Ability in Higher Education

Virawan Amnouychokanant, King Mongkut's University of Technology Thonburi, Thailand
Surapon Boonlue, King Mongkut's University of Technology Thonburi, Thailand
Saranya Chuathong, King Mongkut's University of Technology Thonburi, Thailand
Kuntida Thamwipat, King Mongkut's University of Technology Thonburi, Thailand

The Asian Conference on Education 2020
Official Conference Proceedings

Abstract
The purpose of this research was to identify teaching guidelines to cultivate the computational thinking ability of higher education students. This qualitative research study focused on seven Thai instructors from public and private institutions (Chulalongkorn University, King Mongkut's University of Technology Thonburi, Kasetsart University, Silpakorn University, Assumption University, and Microsoft (Thailand) Limited). All instructors have had teaching experiences in universities for more than five years and some always have used technologies in their classrooms to improve learners' computational thinking ability. Data were collected through instructor focus group interviews. A semi-structured interview protocol was used as a guide. From the interview, we found that three elements for teaching guidelines to cultivate the computational thinking ability of higher education students were 1) learners' and instructors' role 2) learning strategies and 3) teaching tools. The instructor should use learner-centered teaching approaches. In classroom activities, the instructor should be a coach who provides guidance and give powerful questions that help the learners reflect and find a way to get the solution. Besides, this paper gathered learning strategies and teaching tools that were often used in computational thinking courses.

Keywords: Computational Thinking Ability, Instructors Role, Learners Role, Learning Strategies, Teaching Guidelines, Teaching Tools

1. Introduction

The term computational thinking (CT) has been in academic research for decades. In 2006, Jeanette Wing published the viewpoint essay "Computational Thinking" in Communications of the ACM (Wing, 2006) and used CT to describe a set of thinking ability that learners in all fields require to succeed (Czerkawski & Lyman, 2015). Wing's definition of CT ability is proper for application across multiple fields. "Computational thinking is a way that humans solve problems; it is not trying to get humans to think like computers" (Wing, 2006, p. 35). CT is also a key skill for learners in the 21st century (Wing, 2016).

CT has become more important in various fields, and many countries have attempted to integrate CT concepts in other courses (Angeli et al., 2016). For example, the UK has carried out a set of CT courses, including computer science, information technology, and digital literacy throughout all disciplines (Brown, Sentance, Crick, & Humphreys, 2014). Another example is Australia, where CT training was set up as one of the national teaching courses for making the learners familiar with using technology to solve complex problems (Falkner, Vivian, & Falkner, 2014; Armoni, 2012). Poland has also developed computer courses for learners. The main purpose of the development is to help learners understand and analyze the problem, use computers to solve problems, and apply CT to their daily lives (Sysło & Kwiatkowska, 2015).

CT is implemented in courses to train learners' CT ability in many countries. It is tough to imitate CT teaching methods because of the differences in the educational systems and culture (Heintz et al., 2016). However, the key to developing CT is the teachers who have to cultivate their students. Ministry of Education needs to train the teachers in how to design CT learning activities so that the learners can improve their CT systematically and apply CT to other subjects (Orvalho, 2017). Besides, CT also enables learners to become more capable of problem-solving and helps learners develop skills that are attractive for future employment opportunities. (Czerkawski & Lyman, 2015). Computer science is the fast-growing job market and learners who have the ability in coding are highly sought by employers (Dishman, 2016).

For the reasons mentioned above, the purpose of this research was to identify teaching strategies guidelines to foster the computational thinking ability of higher education students.

2. Background

Computational Thinking (CT) is one of the skills that can be useful not only for learners of Computer Science but for other people. CT relates to solving problems, designing systems, and understanding human behavior by connecting the fundamental concepts to computer science (Wing, 2006). In the past studies, CT can be classified into various thinking processes (Table 1), including decomposition, pattern recognition, abstraction, algorithm design, debug and error detection, data collection, data analysis, data representation, automation, simulation, and modeling.

Thinking processes	Explanation	References
Decomposition	Breaking down a complex problem or system into smaller parts that are more manageable and easier to understand.	Hsu, Chang, & Hung, 2018; Curzon et al., 2014; Kazimoglu et al., 2012
Pattern Recognition	Finding the similarities or patterns among small, decomposed problems that can help us solve more complex problems more efficiently.	Hsu, Chang, & Hung, 2018; Kazimoglu et al., 2012; Ismail, Ngah, & Umar, 2010
Abstraction	Focusing on the important information and ignoring unnecessary details.	Grover & Pea, 2013; Wing, 2006
Algorithm Design	Creating a set of step-by-step instructions for solving similar problems or for performing a task.	Mishra & Yadav, 2013; Basu, Biswas, & Kinnebrew, 2017; Choi, Lee, & Lee, 2016
Debug and error detection	Finding mistakes and fix them	Atmatzidou & Demetriadis, 2016; Yadav et al., 2014
Data Collection	Gathering and measuring information from a variety of sources to get a complete and accurate picture of an area of interest.	Rouse, 2020; Barr & Stephenson, 2011
Data Analysis	Inspecting, cleaning, transforming, and modeling data to discover useful information for decision-making.	Choi, Lee, & Lee, 2016; Atmatzidou & Demetriadis, 2016; Angeli et al., 2016; Magana & Silva Coutinho, 2017; Cesar et al., 2017; Basu, Biswas, & Kinnebrew, 2017
Data Representation	Organizing information in the form of graphs, charts, pictures, letters, symbols, and numbers.	Stefan, Gutlerner, Born, & Springer, 2015; Weintrop et al., 2016; Benakli et al., 2017
Automation	Having computers execute repetitive tasks.	Kim, Kwon, & Lee, 2014; Forrest & Mitchell, 2016
Simulation	Using a model to study the performance of a system.	Kim, Kwon, & Lee, 2014; Grover & Pea, 2013; Wing, 2006
Modeling	Creating a model which represents a system including their properties.	Kim, Kwon, & Lee, 2014; Basu, Biswas, & Kinnebrew, 2017; Barr & Stephenson, 2011

Table 1: The classification of CT

Brennan and Resnick (2012) also proposed three dimensions of CT: computational concepts, computational practices, and computational perspectives. See Table 2. Many instructors use programming languages to teach CT although it can be

integrated with various subjects. In facts, CT has been used in different subjects, including mathematics (Snodgrass, Israel, & Reese, 2016; Benakli, Kostadinov, Satyanarayana, & Singh, 2017), biology (Libeskind-Hadas & Bush, 2013; Rubinstein & Chor, 2014), language (Evia, Sharp, & Pérez-Quiñones, 2015), computer science (Shell & Soh, 2013; Grover, Pea, & Cooper, 2015), and programming (Bers, Flannery, Kazakoff, & Sullivan, 2014; Wolz, Stone, Pearson, Pulimood, & Switzer, 2011).

Dimension	Examples
Computational concepts	Sequences
	Loops
	Conditionals
	Events
	Parallelism
	Operators
Computational practices	Incremental and iterative development
	Testing and debugging
	Remixing and reusing
	Abstracting and modularizing
Computational perspectives	Expressing and questioning about the technological world

Table 2: Summary of the CT dimensions

3. Method

3.1 Participants

Seven Thai instructors were invited to take part in the focus group interview. Each lecturer has different proficiencies: CT, coding, and learning strategy. The information of the participants is shown in Table 3.

Gender	Academic position	Workplace	Proficiency		
			CT	Coding	Learning strategy
Male	Assoc.Prof.	Kasetsart University	✓		✓
Male	Assoc.Prof.	Assumption University	✓	✓	✓
Female	Asst.Prof.	Chulalongkorn University			✓
Male	Asst.Prof.	Silpakorn University	✓	✓	✓
Male	Dr.	Microsoft (Thailand) Limited	✓	✓	✓
Male	Dr.	King Mongkut's University of Technology Thonburi	✓	✓	✓
Female	Dr.	Thai MOOC			✓

Table 3: The participants' information

3.2 Instrument

A semi-structured focus group interview was designed for finding teaching strategies guidelines to foster CT ability in higher education. Poorly worded, biased, or

awkward questions can derail a focus group interview and spoil the quality of data. On the other hand, asking good questions makes powerful information so the focus group interview consisted of six open-ended questions which each of them did not ask dichotomous questions (yes or no) and use "think back" questions for taking participants back to their experience. IOC of each item was 0.67 and 1.00 (See Table 4).

Items	IOC
1. What is the role of the instructors in the CT course in higher education?	1.00
2. What is the role of the learners in the CT course in higher education?	1.00
3. Studying in groups, pair, or individual: which way is better to enhance CT?	0.67
4. Which way is better to divide learners into groups (random, up to learners, or different performance-based)?	0.67
5. What learning strategies can be applied for the development of CT in higher education?	1.00
6. What teaching tools can be used to improve CT of learners?	1.00

Table 4: IOC of each item

3.3 Procedure

We divided the procedure of Focus Group Discussion (FGD) into three parts: Before conducting FGD, During FGD, and After FGD.

3.3.1 Before conducting FGD

We designed the opened-ended questions for FGD and reserved the meeting room. After we set the location, date, and time, we sent the invitation letters to experts in computational thinking, coding, and learning strategies. The invitation letter consisted of FGD detail such as topic, venue, date, and time. We chose the location of FGD that is in a convenient place for all participants. We set the duration of the focus group interview one and a half hours. If the FGD is shorter than 60 minutes, it is often difficult to fully explore the discussion topic. If the FGD is longer than 90 minutes, the discussion can become unproductive (as participants get weary).

3.3.2 During FGD

After welcome all participants, we asked them for permission to record the audio during the discussion. One of the researchers was a moderator and the others are note-takers. The moderator allowed all participants to express their opinions and experiences. If someone had given a general answer, the moderator would have asked them to specify by giving an example.

3.3.3 After FGD

Transcribe the audio recorded on the smartphone, cutting out anything unnecessary. Enter the answers to each question into a spreadsheet and begin to analyze the data by organizing the responses into categories. We wrote a report by outlining the major findings and conclusions, as well as the recommendations of participants.

4. Results

From FGD, it can be summarized into various issues to prepare in teaching strategies guidelines to foster the computational thinking ability in higher education.

4.1 The role of the instructors in the CT course in higher education

4.1.1 Knowledge and understanding of CT teaching

The instructors should have experience, expertise, and understanding of CT teaching. Learners can learn from the material at any time, so they are less dependent on the instructor. Teaching materials help learners to learn better. The instructors should act as a coach or facilitator, need not tell everything about the solution, so that learners can solve problems by themselves, leads to systematic thinking. The instructors should connect real-world problems with CT teaching so that learners can understand and apply CT to their real-life more easily. The instructors need to know the different learning styles of learners from several ways such as observation, interview, and questionnaire to properly organize teaching activities.

4.1.2 Preparation and development for CT teaching

CT Training is crucial for CT development. The instructors should realize the importance of CT and always would like to develop themselves to learn new things. Educational institutions should provide CT training for instructors to apply ideas and create new CT instructional materials, which help the teaching and learning to be quality and to increase the interest of learners. With regular CT training, instructors can develop their ability to design learning to enable learners to develop CT sustainably. When learners cannot solve the problems or follow some steps of the process, the instructor must diagnose and guide the way to solve problems. These are the reasons why the instructors must practice or train about CT before teaching.

4.2 The role of the learners in the CT course in higher education

Developing CT ability does not depend on only the instructors but also cooperation from the learners. From FGD, it can be concluded that learners' role in the CT course is to be keen on what they are being taught. The learners need to be active participants in virtually everything that happens in the CT classroom. Learners can help their instructors make decisions such as how a lesson will be delivered or even what is taught. The learners should take responsibility for what is learned and be accountable for the results of the learning process. Their responsibility is demonstrated in their choices and actions, which could lead them to their goal or astray. Therefore, learners should be responsible for everything they are tasked to do by their instructors and attempt to contribute to the CT learning process. Besides, learners should help each

other while working to achieve common learning goals. The learners should find passion in their project or assignment to exceed expectations. Not necessarily go over-the-top, but be able to apply their ability, ask questions, and understand the importance of CT. And most importantly, learners should learn to understand CT and find ways to apply what they have learned in CT class in their daily life, not memorize the CT theory or concept to pass the examinations. To make CT learning effective, learners should make sure they inquire more about particular issues, especially when they feel they need to know more or haven't fully understood.

4.3 Studying in groups, pair, or individual: which way is better to enhance CT?

The instructors need to know the characteristics of the learners (previous experiences of the learners, personal learning styles, cognitive abilities of the learners, personality, aptitude, or intelligence of learners) before choosing the method (studying in groups, pair, or individual). For example, the learners in Mathematics-Science Program can learn by themselves so they like to learn individually while the learners in Language-Arts Program like to learn in a group. Besides, the instructors should pair high and low performers; the learners can learn from their friends. If the instructors would like to group the learners, do not make them more than five people per group because excessive group members can make group work inefficient. The optimal number of members per group should be three to five people. When making the learners into the group, the instructors should let each learner think individually about a topic or answer and then comes back to share ideas with the whole group.

4.4 Which way is better to divide learners into groups (random, up to learners, or different performance-based)?

Each grouping method has its own advantages and disadvantages. For example, if learners can choose their own group, it will make them happy and feel comfortable when working together. On the contrary, high performers will be in the same group. This may make low performers be ignored.

Random Grouping Strategies is a method of teaming learners when grouping is not dependent on factors such as achievement levels or common objectives. This method may make learners excited about member in group, but this method is no clear standard and criteria for grouping.

Different performance-based grouping (high, medium, and low) is reasonable method because it creates learners helping each other within the group to achieve the same goal without ignoring low performers.

4.5 Learning strategies for the development of CT in higher education

Learning strategies are what learners do in their learning process to get a better understanding of the lesson and enhance their own learning. Learning strategies are particularly significant for CT courses because they are tools for active, self-directed involvement, which is essential for developing CT ability. Learners need to use learning strategies as tools to achieve their goals because everything cannot be taught in the class, then learners have to study by themselves. Therefore, learning strategies help learners to study with or without instructors effectively. From FGD, we list the

learning strategies that the instructors have been used for the development of CT. The advantages and disadvantages of each learning strategy are shown in Table 5.

One instructor explained why the instructors should not deliver CT content by using a single method (learners are passively listening):

"When the lecturers read a pre-prepared script with little or no scope for interaction, it makes learners less eager to study. Passively listening to a lecture can be useful at encouraging learning to remember and understand but is not good at encouraging higher-level skills like apply, analyze, and evaluate."

It can be assumed that 'Delivery mode' lectures, where students listen rather than interact, are not good at encouraging higher-level learning and skills.

Learning strategies	Advantages	Disadvantages
Problem-based learning	It is helping learners to improve CT through a problem scene.	Creating suitable problem scenarios is difficult for the instructors and it requires more preparation time.
Project-based learning	Complex tasks allow learners to look at problems with CT, asking questions, and coming up with possible solutions for their project.	It gives a loss of time to the instructors. it also wastes money to buy the supplies for the project.
Game-based learning	The interaction involved in games can help learners understand CT better.	If games are not designed correctly, it could be a disadvantage to the learner's thinking.
Inquiry-based learning	It allows learners to develop CT and research skills. Good questions can open their minds and help develop learners into creative thinkers.	If instructors do not absolutely understand, they are unable to engage with their students on a deeper level.
Scaffolding	It trains the learners to solve problem independently and helps the learners learn the new knowledge.	Instructors are not trained specifically in this method are improbable to deliberately allow learners to make mistakes in the process of learning.
Design-based learning	It helps learners to set up their own goals and to create ideas to achieve them.	It is time-consuming and poses pedagogical challenges.
Digital storytelling	It can help learners practice CT ability. Digital storytelling empowers learners to be confident communicators and creators and reach a deeper	Digital storytelling takes a lot of time to complete the CT project. If the learners had known the assignment at the beginning of the CT course so that they would have had

	understanding of the CT curriculum.	sufficient time to prepare for the assignment. Because of copyright, the learners can not show their real ability and exert their utmost effort only with copyright-free materials.

Table 5: The advantages and disadvantages of each learning strategy

4.6 teaching tools for improving CT of learners

The most teaching tools which the instructors used for designing CT learning activities were block-based programming because most instructors believed that using block in coding can eliminate syntax error which is a barrier for learners to better understand the main programming concepts and block-based programming is suitable for learners who are just starting to practice coding or have little programming experience. It is also found that Scratch is one of the most popular programming languages to learn.

The main reasons why many instructors used Scratch to promote CT are 1. Scratch can be used by people of all ages, including learners from elementary to high school and adults in various settings; 2. Scratch allows users to integrate creativity in storytelling, games, and animation. Learners can collaborate on projects and share their projects online; and 3. Scratch is a free program so people can access and utilize Scratch for both personal and academic use. Apart from Scratch program, it is also having other programming tools for being applied in teaching for cultivating CT to learners such as Alice, LEGO, and code.org (similar to Scratch), etc. Apart from block-based programming, unplugged activities using free exercises from Code.org. This is especially helpful in countries with limited resources, but also in developed countries, where CT is regarded interesting, but there is a lack of resources and experienced instructors.

The instructors also described logical thinking as an integral aspect of CT. One of them stated:

“Whether the learners are giving each other instructions in unplugged activity or creating a game in block-based programming, they are doing it in logical steps and through logical thinking. It makes them more logical for decision-making and problem-solving.”

5. Discussion

This study is conducted to better understand the teaching strategies guidelines to foster computational thinking ability in higher education. The results show that developing CT ability does not depend on only the instructors but also cooperation from the learners, the instructors should have an understanding of CT teaching and they need to practice or train about CT before teaching. They should also connect real-world problems with CT teaching so that students can understand and apply CT to their real-life more easily. In the same way, the learners should be responsible for

everything they are tasked to do by their instructors and attempt to contribute to the CT learning process.

Supportively, results from the past studies reported that the most frequently suggested method for improving CT ability is using real-world problems (Berikan & Özdemir, 2020). It is helping learners to set their own learning goals through a problem in their real-life. Learners will explore the solution by themselves and report their own conclusions to the team. Using real-world problems is not only used to solve problems but also to enhance learners' understanding of computational thinking through appropriate questions (Wood, 2003).

6. Limitations and future studies

Both instructors and students are crucial for CT development. This study collects the instructors' perspectives that may reflect only one side of view. As we work to fill in gaps in understanding and design class activities for our students, future studies should also collect students' views because learners' voices are a powerful tool for CT development.

7. Conclusion

The Focus Group Discussion (FGD) was carried out with seven Thai instructors. The purpose of FGD was to identify teaching strategies guidelines to foster the computational thinking ability of higher education students. A 6-item semi-structured focus group interview was developed and validated. From FGD, it can be summarized into various issues, including the role of the instructors and the learners in the CT course in higher education. The instructors should have an understanding of CT teaching and practice or train about CT before teaching while the learners should be responsible for everything they are tasked to do by their instructors and attempt to contribute to the CT learning process.

This study gathers the learning strategies (advantages and disadvantages) for the development of CT in higher education, including problem-based learning, project-based learning, game-based learning, inquiry-based learning, scaffolding, design-based learning, and digital storytelling. FGD also suggests that using block-based programming is useful for learners who are just starting to practice coding or have little programming experience. Besides, using block in coding can eliminate syntax error which is a barrier for learners to better understand the main programming concepts.

Acknowledgements

The authors would like to express sincere thanks to Petchra Pra Jom Klao Ph.D. Research Scholarship for supporting the expense of research.

References

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., et al. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society, 19*(3), 47-58.

Armoni, M. (2012). Teaching CS in kindergarten: How early can the pipeline begin? *ACM Inroads, 3*(4), 18-19.

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

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *Acm Inroads, 2*(1), 48-54.

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-adapted Interaction, 27*(1), 5-53.

Benakli, N., Kostadinov, B., Satyanarayana, A., & Singh, S. (2017). Introducing computational thinking through hands-on projects using R with applications to calculus, probability and data analysis. *International Journal of Mathematical Education in Science and Technology, 48*(3), 393-427.

Berikan, B., & Özdemir, S. (2020). Investigating "Problem-Solving with Datasets" as an Implementation of Computational Thinking: A Literature Review. *Journal of Educational Computing Research, 58*(2), 502-534.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72*, 145-157.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the annual meeting of the American educational research association, Vancouver, Canada* (pp. 1-25).

Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. *ACM Transactions on Computing Education (TOCE), 14*(2), 9.

Cesar, E., Cortés, A., Espinosa, A., Margalef, T., Moure, J. C., Sikora, A., et al. (2017). Introducing computational thinking, parallel programming and performance engineering in interdisciplinary studies. *Journal of Parallel and Distributed Computing, 105*, 116-126.

Choi, J., Lee, Y., & Lee, E. (2016). Puzzle based algorithm learning for cultivating computational thinking. *Wireless Personal Communications,* 1-15.

Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). *Developing computational thinking in the classroom: a framework.* Retrieved from http://www.chi-med.ac.uk/publicdocs/WP259.pdf on 16 July 2019.

Czerkawski, B. C., & Lyman, E. W. (2015). Exploring Issues About Computational Thinking in Higher Education. *TechTrends, 59*, 57-65.

Dishman, L. (2016). Why Coding Is Still the Most Important Job Skill of the Future. Retrieved from https://www.fastcompany.com/3060883/why-coding-is-the-job-skill-of-the-future-for-everyone on 23 December 2019.

Evia, C., Sharp, M. R., & Pérez-Quiñones, M. A. (2015). Teaching structured authoring and DITA through rhetorical and computational thinking. *IEEE Transactions on Professional Communication, 58*(3), 328-343.

Falkner, K., Vivian, R., & Falkner, N. (2014, January). The Australian digital technologies curriculum: Challenge and opportunity. *Proceedings of the sixteenth Australasian computing education conference: 148*, (pp. 3-12). Australian Computer Society, Inc.

Forrest, S., & Mitchell, M. (2016). Adaptive computation: The multidisciplinary legacy of John H. Holland. *Communications of the ACM, 59*(8), 58-63.

Grover, S., & Pea, R. (2013). Computational thinking in K–12 a review of the state of the field. *Educational Researcher, 42*(1), 38-43.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education, 25*(2), 199-237.

Heintz, F., Mannila, L., & Färnqvist, T. (2016, October). A review of models for introducing computational thinking, computer science and computing in K-12 education. *Frontiers in education conference (FIE), 2016 IEEE* (pp. 1-9). IEEE.

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education, 126*, 299-300.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010). The effects of mind mapping with cooperative learning on programing performance, problem solving skill and metacognitive knowledge among computer science students. *Journal of Educational Computing Research, 42*(1), 35–61.

Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play. *Procedia Computer Science, 9*, 522-531.

Kim, Y. C., Kwon, D. Y., & Lee, W. G. (2014). Computational modeling and simulation for learning an automation concept in programming course. *International Journal of Computer Theory and Engineering, 6*(4), 341-345.

Libeskind-Hadas, R., & Bush, E. (2013). A first course in computing with applications to biology. *Briefings in Bioinformatics, 14*(5), 610-617.

Magana, A. J., & Silva Coutinho, G. (2017). Modeling and simulation practices for a computational thinking-enabled engineering workforce. *Computer Applications in Engineering Education, 25*(1), 62-78.

Mishra, P., & Yadav, A. (2013). Of art and algorithms: Rethinking technology & creativity in the 21st century. *TechTrends, 57*(3), 10-14.

Orvalho, J. (2017, July). Computational thinking for teacher education. *Scratch2017BDX: Opening, inspiring, connecting* (pp. 6).

Rouse, M. (2020). Data Collection. Retrieved from https://searchcio.techtarget.com/definition/data-collection on 14 October 2020.
Rubinstein, A., & Chor, B. (2014). Computational thinking in life science education. *PLoS Computational Biology, 10*(11), e1003897.

Shell, D. F., & Soh, L. K. (2013). Profiles of motivated self-regulation in college computer science courses: Differences in major versus required non-major courses. *Journal of Science Education and Technology, 22*(6), 899-913.

Snodgrass, M. R., Israel, M., & Reese, G. C. (2016). Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis. *Computers & Education, 100*, 1-17.

Stefan, M. I., Gutlerner, J. L., Born, R. T., & Springer, M. (2015). The quantitative methods boot camp: Teaching quantitative thinking and computing skills to graduate students in the life sciences. *PLoS Computational Biology, 11*(4), e1004208.

Sysło, M. M., & Kwiatkowska, A. B. (2015, September). Introducing a new computer science curriculum for all school levels in Poland. *International conference on informatics in Schools: Situation, evolution, and perspectives* (pp. 141-154). Cham: Springer.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., et al. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127-147.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2016). *Computational thinking, 10 years later*. http://www.microsoft.com/enus/research/blog/computational-thinking-10-years-later.

Wolz, U., Stone, M., Pearson, K., Pulimood, S. M., & Switzer, M. (2011). Computational thinking and expository writing in the middle school. *ACM Transactions on Computing Education (TOCE), 11*(2), 9.

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

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE), 14*(1), 5.