DOI: 10.34641/ctestem.2022.467

How the Pre-service Teachers Associate Computational Thinking with Programming? A Case Study of an Introductory Programming Course in Teacher Education

Megumi IWATA[1*], Jari LARU[2*], Kati MÄKITALO[3*]
[1,2,3] Faculty of Education, University of Oulu, Finland
megumi.iwata@oulu.fi, jari.laru@oulu.fi, kati.makitalo@oulu.fi

ABSTRACT

There is a growing interest among educators to integrate computational thinking into basic education. Computational thinking is a complex concept and difficult to understand especially for those who have limited theoretical knowledge about this concept and no background in the computer science. Question arises, whether we reach the high-standard learning goals without comprehensive understanding of computer science. Therefore, there is a need to study computational thinking and how it should be introduced to pre-service teachers with little knowledge and experience in computer science and programming. This study aims to explore pre-service teachers' understanding of computational thinking in the context of an introductory programming course. We focus on to what extent the pre-service teachers recognize computational thinking during the course and how they associate their conceptual understanding of computational thinking with the concrete programming practices. We undertake in-depth analysis of five pre-service teachers who were novices in programming. The assignments and the survey after the course are analysed. The preliminary results show that sequencing from unplugged activities to computerized activities and project work helps the pre-service teachers recognized computational thinking. Understanding of the relationship between computational thinking and programming was diverse. Some explained that computational thinking helps understanding the code. This study provides insights of how computational thinking should be introduced along the way of learning programming.

KEYWORDS

Computational Thinking, Programming, Teacher Education, K-12 Education, Case Study

1. INTRODUCTION

Programming is a difficult subject for novices. Selby (2015) explains the learning difficulties of programming, which indicates the lack of ability to understand how a computational model works, to master reading, tracing and writing code, and to understand high-level concepts, such as design. Learning programming requires thinking and metacognitive skills, knowledge and information from multidisciplinary fields (e.g., Durak, Yilmaz, & Yilmaz, 2019; Selby, 2015; Li, 2016).

Wing (2006) states CT is a skillset for everyone. She defines that CT "is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Cuny, Snyder, & Wing, 2010, as cited in Wing, 2010, p.1). Later Aho (2012) redefines CT as the thought processes involved in formulating problems so that "their solutions can be represented as computational steps and algorithms" (p. 832).

While programming is a process to solve certain problems with computing, CT involves not only programming but also views which reflect benefits provided by computing for solving problems (Kukul & Karatas, 2019). Indeed Wing (2006) claims that CT is not programming but more like conceptualizing. Understanding such concept may be more challenging for those who have limited knowledge in computer science and experience in computer programming.

To advance CT education, developing teachers' ability is considered as a key factor (Kong, Abelson, & Lai, 2019). In teacher education, teachers not only obtain programming skills, but also understand CT and practice pedagogy to effectively foster CT. Despite the increasing interest in CT and programming among educators, the previous study found that many teachers had little understanding of CT skills and how they could develop CT skills in the practice (Sands, Yadav, & Good 2018). There is a need to develop teacher education to equip pre-service teachers with ability to think computationally to be able to integrate CT into education (Yadav, Gretter, Good, & McLean, 2017).

This study aims to explore how the pre-service teachers associate CT with programming in the context of an introductory programming course in teacher education. There has been less emphasis on how CT can help learning programming compared to how CT can be developed through programming. To achieve the aim, we set the following research questions: 1) To what extent do the pre-service teachers recognize CT in the introductory programming course? 2) In what ways do the pre-service teachers associate CT with programming?

With the first research question we investigate pre-service teachers' perceptions about CT in the introductory programming course. With the second research question, we aim to understand how the pre-service teachers relate conceptual CT with practical programming. The findings of the study can provide views on how CT should be introduced for programming novices along the way of learning programming.

2. PROGRAMMING AND CT

Previous literature review revealed the distinct emphasis on programming among the CT related literature (Saqr, Ng, Oyelere, & Tedre, 2021). Moreno-León, Román-González, and Robles (2018) claims that there are two main strategies to develop CT: unplugged activities and computerized activities. Unplugged activities include logic games, cards, puzzles to get to know computer science concepts.

Appropriate unplugged approach may help learning process of novices. For instance, Looi, How, Longkai, Seow, and Liu (2018) conclude that unplugged activities can possibly help understand key concepts of computing and develop CT.

Computerized activities, such as programming, expose students to CT using computer sciences concepts (Lye, & Koh, 2014). Dural and colleagues (2019) point out that thinking skills and knowledge in different fields required in the processes is considered as a basic strategy for developing CT and computer-based problem-solving process. Li (2016) suggests the close relationship between CT and programming course. CT should be the goal for the programming course because the focus is broader, problem-solving, and thinking skills not limited to programming language. The programming course can provide a practical carrier to the cultivation of CT ability because 1) programming is the best way to express CT, 2) programming course may include thinking methods of CT and 3) practices in programming course can provide opportunity to train CT (Li, 2016). Inquiry-based pedagogical approach includes problem solving and requires thinking skills, creativity and provides the platform for adapting theory to practise (Häkkinen, Järvelä, Mäkitalo-Siegl, Ahonen, Näykki, & Valtonen, 2017, Iwata, Laru, & Mäkitalo, 2020).

3. METHODS

This is a case study in which we explore pre-service teachers' experiences in an introductory programming course.

3.1. Participants

The participants are five pre-service teachers (Pre-service teacher A-E), who participated in the course (see Table 1). Pre-service teacher A, B and C were from primary teacher education program, and Pre-service teacher D and E were from subject teacher education program.

Table 1. Demographics of the Participants.

Pre-service teacher	Study in university	Teaching experience	Programming experience
A	1 year	None	None
B	1 year	None	None
C	3 years	1 year	None
D	4 years	1 year	None
E	1 year	1 year	None

3.2. Introductory Programming Course

The introductory course entitled *"programming in basic education"* is an optional course at the pre-service teacher education. This course corresponds to 5 ECTS[1] and is estimated as 133.5 hours of study including 20 hours of lectures, 30 hours of exercises, as well as self-study.

The main contents of the course included: 1) familiarizing oneself with collaborative problem-solving in the context of programming, 2) familiarizing oneself with the contents related to programming in the basic education curriculum, 3) practicing the basics of computational thinking, 4) getting to know different programming tools and environments for beginners, and 5) understanding the basics of automation with robotics tools. The tools used in the course were divided into five topics: 1) unplugged programming, 2) visual programming, 3) tinkering, 4) programming for games, and 5) robotics. In the spring 2021, 12 pre-service teachers participated in the course, which was organized as mixture of distance and face-to-face lessons (hybrid learning) because of the covid-19.

In the course, collaborative inquiry learning method was used as an approach to provide pre-service teachers experience on this kind of pedagogy. Assignments (group or individual) were given in each topic. Examples of the assignments were: *Create a coding project using the tool; Plan a small learning activity using the coding tool.* Pre-service teachers engaged in the project work, where they created learning materials for robotics programming activities with BBC micro:bits. The pre-service teachers were divided into two groups and created the learning materials consisted of multiple programming tasks. Pedagogically they were challenged by adding structure to adjust difficulties, examples and hints to help students proceed, and guiding questions to encourage reflection.

CT was introduced to the pre-service teachers in the beginning of the course and pointed out throughout the course along with learning of different topics. The programming skills by the National New Literacy Development Program *(Uudet Lukutaidot[2])* was used as a main framework in this course. Uudet lukutaidot is a joint program of the National Audiovisual Institute and the Ministry of Education in Finland. The framework describes programming related skills in three categories and nine subcategories. CT is one of the categories, which includes the four subcategories: logical thinking and information processing, problem solving and modelling, programming concepts and structure, and programming practices. This framework was chosen because it provides practical information for the teachers such as age-appropriate pedagogy and instructions. In addition, Brennan and Resnick's (2012) three dimensions of CT elements were explained by the teacher. The frameworks were provided as a foundation for the pre-service teachers so that they can recognize and practice CT by themselves while working on the course assignments and the project.

3.3. Data Collection and Analysis

The data for this study includes a survey filled by the pre-service teachers after the course and the assignments and the materials produced during the course. The survey included 16 questions about their experience during the course. CT was mentioned in the survey questions, such as "*CT was clearly part of this course*" and "*I understand how programming and CT are related".* The survey questions were answered with the five-point Likert scale followed by the further questions to ask the reasons of the choice.

The data was in Finnish which was later translated into English. The data was analysed inductively. First, the researchers read the data and familiarized themselves with the data. Then the researchers marked the parts of the data which were related to the research focus of the study. Those

[1] European Credit Transfer and Accumulation System (ECTS)

[2] https://uudetlukutaidot.fi/ohjelmointiosaaminen/

parts were categorized by themes. Processes of modifying the themes and dividing the data into themes were repeated.

4. RESULTS

4.1. Learning CT through Practicing Programming

Four out of five pre-service teachers agree that CT was clearly part of the course. However, the results indicate that pre-service teachers perceive CT differently in the course.

In the survey, two pre-service teachers indicate that CT was well visible in the topic of unplugged programming. Various activities and web resources for unplugged programming were presented to the pre-service teachers in the beginning of the course. The answers in the survey indicate that the structure, which starts with unplugged exercises and continues with visual programming, robotics and project work, can deepen understanding of CT. Two pre-service teachers' answers in the survey are as follows.

I think computational thinking was visible throughout the course. The course began with unplugged programming, which led to connecting with computational thinking. The exercises in the course were multifaced and developed computational thinking in different ways. For example, nice board games and apps (Scratch Jr + Scratch) led to solving problems piece by piece. (Pre-service teacher A)

I think computational thinking came up right at the beginning of the course when we program each other like a robot (unplugged). Immediately, such exercises provoked to think about computational thinking, which we then deepened through games, robotics, and project work. (Pre-service teacher D)

The assignment, where the pre-service teachers reflected on how CT was visible in the unplugged activities, shows the their understanding of CT and unplugged programming. Pre-service teacher A answered in the assignment as follows: "the student creates step-by-step instructions using simple commands and a repeat structure"; "the student recognizes the errors in the instructions and tries out solutions to correct them"; "the student develops precise and detailed instructions for using repeat and conditional structures".

The project work was explorative and ill-structured problem solving, where the pre-service teachers may apply CT. Pre-service teacher E expresses that she recognized CT in the problem-solving process during the project work.

Producing the content of project work required computational thinking; in particular, the content team had to think and come up with a wide range of problems and tasks, assess their difficulties, arrange these challenges to create meaningful entities, and consider possible different initial levels [of programming] to find meaningful things for everyone. When doing things, I did not notice, but after looking at it, I can see how the thought process has progressed and find the features of computational thinking there. (Pre-service teacher E)

The results indicate that pre-service teachers have various levels of understanding of CT through programming practices. The below quote shows that pre-service teacher C think that CT was not visible enough in the course.

In my opinion, the tasks and exercises of the course "forced" a different way of thinking and helped to develop computational thinking. However, there was little emphasis on thinking in the lessons, for example, and the perception of such thinking was not noticed until after the course. (Pre-service teacher C)

4.2. Relationship between Programming and CT

All five pre-service teachers show confident in understanding the relationship between programming and CT. Pre-service teacher D and E state that CT is behind practices in programming, such as problem solving, logical thinking and creative process. Pre-service teacher D explains that "It [CT] is about thinking, developing, problem solving like an IT expert or a computer would do. When you program, you get a certain kind of 'sense of control' about creating something new, more effective, and meaningful". Pre-service teacher E explains the connections between CT and programming practices as below.

Computational thinking is part of programming. It involves basic notions of programming, logical reasoning, and problem solving. Computational thinking is behind all programming activities, influencing action, thinking, and creation. Understanding a problem, finding a solution to it, and putting the solution into practice are all computational thinking and its outcome. Computational thinking thus serves as a kind of basis for all other programming activities. (Pre-service teacher E)

Two pre-service teachers mention that CT can be developed by programming. "Computational thinking can be taught through programming, for example, engaging in unplugged programming or programming with devices allows you to practice and develop computational thinking skills" (Pre-service teacher A).

Pre-service teacher B and C explain that CT helps to understand programming. With understanding of CT and programming concepts, the pre-service teachers may better understand programming practices including the meaning of the code. Pre-service teacher B wrote in the survey that "recognizing sequences and understanding the purpose of commands, these aspects combine computational thinking and programming" (Pre-service teacher B). Pre-service teacher C explained in the survey as follows.

Computational thinking allows general understanding of programming and makes it easier to understand how programming works. In particular, computational thinking is emphasized when looking at, for example, the operation and meaning of commands in programming. A logical mindset and the ability to perceive repetitive "rules" make it easier to understand how programming works. (Pre-service teacher C)

The assignments to read and remix others' code gave the pre-service teachers opportunity to practice using CT as a help to understand the code. In the assignment of visual programming, the pre-service teachers remixed existing Scratch projects. Using a Scratch game that adds a point when a character is clicked, pre-service teacher B remixed the game by adding a new character that reduces a point when it is clicked. In addition, she made two characters have conversation. To do so, she needed to understand the code of the original character and make modifications in the code.

These are the preliminary results, which indicates that the pre-service teachers understood the meaning of CT and programming practices differently through this course. Further, more detailed perception on how pre-service teachers built their understanding about CT through different exercises will be presented at the conference.

5. DISCUSSION AND CONCLUSION

This study aims to explore how pre-service teachers associate CT with programming in the context of the introductory programming course. The course provided opportunities to practice CT through different programming assignments. Such opportunities can cultivate CT (Li, 2016) and encourage to think computationally, which is the first step for pre-service teachers to integrate CT (Yadav et al., 2017). We note three main findings from the preliminary results that can be used to improve the approaches to introducing CT along the way of learning programming. First, the pre-service teachers' perceptions on how CT relates with programming differ. One of the reasons is that relationship between CT and programming was not explained explicitly by the teacher but the pre-service teachers were expected to build understanding by themselves. Second, the results indicate that sequencing the learning topics from unplugged activities to computerized activities and project work, where all these learnt issues must be adapted, helps pre-service teachers to understand programming and acknowledge CT in relation to programming practices and a bigger picture of problem solving. Third, the results demonstrate that CT can help to understand programming. The pre-service teachers used CT to overcome the inability that causes the difficulties of learning programming, such as understanding how a computational model works, and mastering reading, tracing, and writing code, as Selby (2015) described.

As limitation of the study, we acknowledge that the number of the participants are small. We tried to understand the pre-service teachers who were novices in programming with multiple data sources. In future, interview methods and pre- and post-assessments of CT may give deeper insights of the pre-service teachers' understanding of CT. Pedagogical aspects should be explored more in the future studies, which addresses diverse ways to teach unplugged activities, games, and robotics in pre-service teacher education, to find out efficient approaches for learning CT and programming.

6. REFERENCES

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835.

Durak, H. Y., Yilmaz, F. G. K., & Yilmaz, R. (2019). Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. *Contemporary Educational Technology, 10*(2), 173-197.

Häkkinen, P., Järvelä, S., Mäkitalo-Siegl, K., Ahonen, A. K., Näykki, P., & Valtonen, T. (2017). Preparing teacher students for twenty-first-century learning practices: a framework for enhancing collaborative problem-solving and strategic learning skills. Teachers and Teaching: Theory and Practice, 23(1), 25-41.

Iwata, M. Laru, J., & Mäkitalo, K. (2020). Designing problem-based learning to develop computational thinking in the context of K-12 maker education. K. Kori K. & M. Laanpere (Eds.), proceedings of International Conference on Informatics in School: Situation, Evaluation and Perspectives, 103-106.

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

Kong, S. C., Abelson, H., & Lai, M. (2019). Introduction to Computational Thinking Education. In Kong, S. C. & Abelson, H. (eds). *Computational Thinking Education*. Singapore: Springer.

Kukul, V., & Karatas, S. (2019). Computational thinking self-efficacy scale: Development, validity and reliability. *Informatics in Education, 18*(1), 151-164.

Li, Y. (2016, October 12-15). Teaching programming based on Computational Thinking. In *2016 IEEE Frontiers in Education Conference*, Erie, PA, 1-7. IEEE.

Looi, C. K., How, M. L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255-279.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior, 41*, 51-61.

Moreno-León, J., Román-González, M., & Robles, G. (2018, April). On computational thinking as a universal skill: A review of the latest research on this ability. In *2018 IEEE Global Engineering Education Conference* (EDUCON) 1684-1689. IEEE.

Sands P., Yadav A., Good J. (2018) Computational Thinking in K-12: In-service Teacher Perceptions of Computational Thinking. In Khine M. (eds) *Computational Thinking in the STEM Disciplines*. Springer, Cham.

Saqr, M., Ng, K., Oyelere, S. S., & Tedre, M. (2021). People, ideas, milestones: a scientometric study of computational thinking. *ACM Transactions on Computing Education (TOCE), 21*(3), 1-17.

Selby, C. C. (2015, November). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. In *Proceedings of the workshop in primary and secondary computing education,* 80-87.

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

Wing, J. M. (2010). *Computational Thinking: What and Why?* The Link Magazine. Retrieved May 4, 2022, from https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In *Emerging research, practice, and policy on computational thinking,* 205-220. Springer, Cham.