Journal Pre-proof

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy

Xuefeng Wei, Ph.D, Lin Lin, Ed.D, Nanxi Meng, Wei Tan, Siu-Cheung Kong, Ph.D, Kinshuk, Ph.D

PII: S0360-1315(20)30221-9

DOI: https://doi.org/10.1016/j.compedu.2020.104023

Reference: CAE 104023

To appear in: *Computers & Education*

Received Date: 13 September 2019

Revised Date: 24 July 2020

Accepted Date: 16 September 2020

Please cite this article as: Wei X., Lin L., Meng N., Tan W., Kong S.-C. & Kinshuk The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy *Computers & Education*, https://doi.org/10.1016/j.compedu.2020.104023.

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy

Xuefeng Wei*, Ph.D.
Professor, School of Teacher Education, Ludong University
186 Hongqi Road, Yantai City, Shandong Prov., China
Email: xuefengwei99@163.com

Lin Lin, Ed.D.
Professor, Department of Learning Technologies, University of North Texas
3940 North Elm Street, Denton, TX 76207, USA
Email: Lin.Lin@unt.edu

Nanxi Meng
Department of Learning Technologies, University of North Texas
1155 Union Circle #311127, Denton, Texas 76203, USA
Email: nanxi.meng@unt.edu

Wei Tan
The experimental primary school of Yantai Economic & Technological Development Area
Xiaoqinghe Road, Yantai Economic & Technological Development Area,
Shandong Province, China
Email: tanweildu@163.com

Siu-Cheung Kong, Ph.D.
Professor, Centre for Learning, Teaching and Technology
The Education University of Hong Kong
10 Lo Ping Road, Tai Po, New Territories, Hong Kong
Email: sckong@eduhk.hk

Kinshuk, Ph.D.
Professor and Dean, College of Information, University of North Texas
3940 North Elm Street, Denton, TX 76207, USA
Email: kinshuk@unt.edu

Dr. Xuefeng Wei is corresponding author of this manuscript.

Abstract

Computational thinking (CT) skills are becoming essential in all aspects of work and life, and CT has become part of the K-12 curriculum around the world. Yet, more research is needed to better understand how to help elementary school students acquire CT skills effectively. The purpose of this study was to examine the effectiveness of partial pair programming (PPP) on elementary school students' CT skills and self-efficacy (SE). The study included four 4th grade classes, with a total of 171 students, who were taking the course entitled "Computational Thinking with Scratch" with the same teacher for one semester. Two classes (84 students) were in the Experimental Group (EG, 44 boys and 40 girls) while two other classes (87 students) were in the Control Group (CG, 45 boys and 42 girls). The students in the EG were paired up for the programming assignments while the students in the CG did the assignments on their own. The results showed that the students in the EG improved their CT skills and SE more significantly than those in the CG group. There was a low correlation between CT and SE in the post-test of both groups. The interviews with the teacher and the students afterwards provided insights in understanding the results. The findings suggest that in K-12 programming teachers could use PPP as an effective approach to improve students' CT skills, programming SE, and beyond.

Keywords: Elementary education; Classroom teaching; Computational thinking; Partial pair programming; Self-efficacy; Pair collaborative learning

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Funding: This work was supported by the National Social Science Foundation of China "Research on the Evaluation System of individualization-oriented Internet Education Services in Primary and Middle Schools" (NO: BCA170075)

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy

Abstract

Computational thinking (CT) skills are becoming essential in all aspects of work and life, and CT has become part of the K-12 curriculum around the world. Yet, more research is needed to better understand how to help elementary school students acquire CT skills effectively. The purpose of this study was to examine the effectiveness of partial pair programming (PPP) on elementary school students' CT skills and self-efficacy (SE). The study included four 4th grade classes, with a total of 171 students, who were taking the course entitled "Computational Thinking with Scratch" with the same teacher for one semester. Two classes (84 students) were in the Experimental Group (EG, 44 boys and 40 girls) while two other classes (87 students) were in the Control Group (CG, 45 boys and 42 girls). The students in the EG were paired up for the programming assignments while the students in the CG did the assignments on their own. The results showed that the students in the EG improved their CT skills and SE more significantly than those in the CG group. There was a low correlation between CT and SE in the post-test of both groups. The interviews with the teacher and the students afterwards provided insights in understanding the results. The findings suggest that in K-12 programming teachers could use PPP as an effective approach to improve students' CT skills, programming SE, and beyond.

Keywords: Elementary education; Classroom teaching; Computational thinking; Partial pair programming; Self-efficacy; Pair collaborative learning

1. Introduction

Computational thinking (CT) skills are as important as the other essential skills such as reading, writing and arithmetic. CT involves a wide range of intellectual processes such as logical thinking, algorithm selection, and system thinking (Tsai, Wang & Hsu, 2019; Wing, 2006). Lee (2011) explored the development of CT skills and identified five CT characteristics. These include conditional logic, algorithm building, debugging, simulation, and distributed computation. Computational programming has been described as a way to analyze and solve problems (Sarpong, Arthur & Amoako, 2013; Knuth, 2014). Due to its practical nature, CT skills are not only important to computational professionals but also essential for future citizens as daily life skills (Wing, 2006). Therefore, it is crucial to help K-12 students develop CT skills and prepare them with problem solving skills for the future.

Programming and CT were traditionally studied among college students and in computer science courses. Yet, in recent years, more attention has been devoted to developing CT skills in K-12 contexts, and these skills have been increasingly added in the K-12 school curriculum. For instance, the American International Educational Technology Association (ISTE, 2016) revised the “Technical Standards for Students” for the developmental needs of students in the digital age. The Standards increased the requirements for cultivating CT, emphasizing the skills of data analysis, modeling and abstraction, algorithmic thinking, and emphasizing skills to solve problems via technical tools, decomposing problems, extracting key information, and modeling for complex systems. The Australian Curriculum Assessment and Reporting Authority (2015) also highlighted the importance of developing students' CT skills, requiring students to analyze data, decompose problems, and applying algorithmic thinking to solve problems with digital tools. The Chinese High School Information Technology Curriculum Standards (2016) proposed creating information technology classes for K-12 students.

Given the importance of programming and CT skills, educational researchers have explored the pedagogical strategies to teach programming and improve CT skills. Partial pair programming (PPP), modified based on pair programming as one of the collaborative learning approaches, has been applied and researched in programming education in colleges and other post-secondary contexts (Lye & Koh, 2014). However, the effect of PPP on self-efficacy (SE) has not been studied among elementary students in detail. In the current study, we propose to teach coding to 4th grade students using the PPP approach, and explore the extent to which PPP might improve the students' CT skills as compared to having students code on their own. In addition, we assessed the students' SE to better understand the relationships between CT skill development, PPP, and the students' personal beliefs and efforts in programming and CT. In the following sections, we discuss the related literature and research gaps in CT and SE. We then introduce our research design and findings. Finally, we draw some conclusions, state some implications, and point out possible avenues for future research.

2. Literature Review

CT encompasses thought processes involved in solving problems, designing systems, and understanding human behavior based on the fundamentals of computer science (Wing, 2012). In education, fostering students' CT helps them obtain higher-level thinking processes such as deciphering problems and forming innovative solutions (Barr, Harrison & Conery, 2011). CT can be cultivated and improved through various subject areas, including subjects widely accepted as CT-intensive such as programming, mathematics (Benakli, Kostadinov, Satyanarayana & Singh, 2017; Weintrop, 2016), biology (Rubinstein & Chor, 2014), robotics (Atmatzidou &

Demetriadis, 2016), and physics (Weintrop et al., 2016). CT can also be cultivated through subjects such as music (Bell & Bell, 2018), business management (Friedman & Jacobson, 2018), and language development (Evia, Sharp & Pérez-Quiñones, 2015). In a K-12 context, programming has been a major tool to help students improve CT skills (Hsu, Chang & Hung, 2018; Lockwood & Mooney, 2017). Programming allows students to think and act through abstraction, generalization, algorithmic thinking and designing, and debugging and iteration (Shute et al., 2017).

2.1 Pair Programming and Partial Pair Programming in K-12 Contexts

Among different pedagogical strategies, collaborative learning has been advocated as an effective approach to help student obtain CT skills (Hanks, 2008). Collaborative learning involves groups of students working interactively to solve a problem, complete a task, or create a product through the sharing of information and the negotiation of meaning relevant to the problem-solving task (Gerlach, 1994; Dillenbourg, 1999; Roschelle & Teasley, 1995). Studies have shown that students learning programming with a collaborative approach perform better in completing programing tasks and improving CT than students programming solely on their own (Zhong, Wang & Chen, 2016).

Among pedagogical approaches to collaborative programming, pair programming (PP) is shown to be an effective strategy to teach programming. PP involves a learning process in which two students collaboratively and simultaneously work on one computer to complete a programming task (Williams et al., 2002). The roles of the two partners in PP typically include those of the driver and the navigator (Williams & Kessler, 2002), and the partners switch roles periodically. The driver of the pair is the operator of the programming actions, who usually types on the computer or writes down a design. The navigator has a more objective point of view. The navigator observes the driver’s work, and provides the strategic and longer-term thinking for the direction of the work. PP benefits programming in several aspects: 1) it improves programming skills and program quality of students due to improved comprehension through the sharing of information and the negotiation of meaning during the learning process (Dongo, Reed & O'Hara, 2016; Li, Plaue & Kraemer, 2013; Liebenberg, Mentz & Breed, 2012; Braught, Eby & Wahls, 2008); 2) it reduces frustration and anxiety of beginning programming students, helps foster positive attitudes in programming, and brings students the enjoyment and sense of achievement (Dongo, Reed & O'Hara, 2016; Liebenberg, Mentz & Breed, 2012; Nagappan et al., 2003); and 3) it helps increase motivation and retention of learning to program (Yang, Lee & Chang, 2016; McDowell, Werner, Bullock & Fernald, 2006). Furthermore, PP has also been observed to improve CT skills (Denner, Werner, Campe & Ortiz, 2014), communicational skills (D'Angelo & Begel, 2017; Nagappan et al., 2003; Sfetsos, Stamelos, Angelis & Deligiannis, 2006), and collaboration skills (Lewis, 2011). Although studies on PP have highlighted the positive effect, experiments and empirical research on PP have heavily focused on tertiary level students and adult learners. Student in the K-12 context, especially elementary school learners are rarely studied (Iskrenovic-Momcilovic, 2019). Zhong, Wang and Chen (2016) studied the social factors of PP among elementary school students and found that working in pairs contributed significantly to programming learning results and achievements. Also, PP helped tighten the partnership within the pairs. Zhong, Wang, Chen and Li (2017) studied the behavior in role switching in four classes of sixth graders in an elementary school. They found that allowing students to switch roles of driver and navigator during one task or class period improved students’ enjoyment of the tasks. Iskrenovic-Momcilovic (2019) studied an elementary programming class using Scratch, and found that students programmed more effectively when programming in pairs than working individually. PP helped motivate the students to acquire new programming language and memorize new concepts faster.

However, research has also shown the limitation of applying PP among young students and beginning programming learners. Chong & Hurlbutt (2007) found that the less experienced partner in PP tends to disengage and learn less. In the study done by Denner et al (2019), the PP students showed constraints interacting with their partners who were less interested or less willing to collaborate. It is worth investigating further how to better design PP to maximize learning results for young students in a given context.

In this study we modified the PP approach to partial pair programming (PPP). We were interested in facilitating a collaborative pair programming environment. Yet, it will be difficult for a teacher to implement the standard PP in a typical 4th grade classroom with over forty pupils, that is, to monitor the regular switching between the roles of driver and navigator within the pair of young students, and to ensure the fairness when assessing students' work. As such, instead of emphasizing the clear role divisions and ask pupils to switch roles regularly, we focused on the collaborative learning aspect of pair programming by asking the paired students to work collaboratively while submitting their individual programming projects. We expect that this will help the pupils to work collaboratively and still have ownership of their own projects. We will include more details of the experiments in the methodology section.

2.2 Using Scratch to Develop Programming and Computational Thinking Skills

Various technologies have been developed to teach students CT skills in K-12 contexts. One of the most popular tools is Scratch (https://scratch.mit.edu/) designed by MIT Media lab. Scratch is a free, easy-to-use visual programming language. With dragging, dropping and snapping of programming blocks, learners can create animations, games, and interactive media projects (Resnick et al., 2009). They can also share these projects in the Scratch community with users from all over the world (Brennan & Resnick, 2012). Scratch helps young learners to develop their critical thinking, systematic reasoning, and collaborative working skills (Lockwood & Mooney, 2017). Studies have shown the effectiveness of using Scratch as a CT learning and improvement tool (Pérez-Marín, Hijón-Neira, Bacelo & Pizarro, 2018).

Brennan and Resnick (2012) studied the Scratch online community and described three key dimensions of CT cultivation: namely, CT concepts, computational practices, and computational perspectives. The CT concepts involve knowledge that the designers employ as they program, i.e. loops, conditions, and so forth. When students program with Scratch, they drag, drop and snap blocks to combine different components in order to complete the task. They select the element, action and number of times the action should be repeated, then they achieve and watch the loop of repeated action. Computational practices refer to the activities that the designers develop as they program, i.e. abstracting and modelling, testing and debugging. The computational perspectives are points of views that the designers form about the world around them and about themselves, i.e. examining the relationship between themselves and the technological world.

Scratch has established itself as a proper tool for teaching and evaluating students' CT skills. Yet, to have a full understanding of how students learn to program, it is also important to understand their personal beliefs and efforts as regards their CT skills and SE.

2.3 Self-Efficacy in K-12 Contexts

Self-efficacy (SE) refers to the psychological state of individuals and their beliefs in themselves and how to relate to different situations (Bandura, 1994). SE affects the amount of effort students demonstrate in various learning contexts, and motivates students to achieve better learning results (Gandhi & Varma, 2009). Students with high SE tend to consider difficult tasks as challenges that need to be addressed rather than things to be avoided (Bandura 1994).

SE has been proven to be an effective tool to reveal the close relationship between students' beliefs in themselves and their academic achievements like reasoning and problem-solving skills (Psycharis & Kallia, 2017). Furthermore, it has been argued to enhance learning performance on specific skills like dance performance skills (Hsia, Huang & Hwang, 2016) and communication skills (Boissy, 2016). In the context of K-12 STEM education, Leonard et al. (2016) found a positive correlation between SE and performance of scientific inquiry. Ketelhut (2010) created for middle school students an instrument called Self-Efficacy in Technology and Science (SETS), which has been adopted by studies in K-12 STEM education. In the context of programming education, SE has shown to have an intimate relationship between oneself and one's actual performance (Kong, Chiu & Lai, 2018). Ramalingan, LaBelle & Wiedenbeck (2004) argued that programming influences students' SE, and SE affects learning performance positively. In the context of K-12 programming education, the relationship between CT skills and SE can also be seen in much established scholarly work. For instance, with greater SE, students showed more confidence in performing the given task, and were more likely to continue working to complete the task (Kong, Chiu & Lai, 2018). Studies also found that with good teaching strategies, learning how program was more likely to be successful, and such learning increased students' SE toward programming (Denner, Werner, Campe & Ortiz, 2014). Among different pedagogical strategies, Yildiz Durak (2018) found that teaching students to develop digital story design using Scratch improved the students' level of understanding programming concepts and programming SE. Kong, Chiu and Lai (2018) found that students with positive attitudes towards collaboration had greater creative SE in programming with Scratch, especially among primary school students.

Additionally, pairing up students when conducting programming tasks greatly improved students' programming SE (Zhong, Wang & Chen, 2016).

2.4 Self-Efficacy, Computational Thinking and K-12 Programming

SE is the foundation of human motivation. It critically contributes to improving performance and one's emotional state (Bandura, 1997, 2006). Without people believing that their actions can achieve their desired effects, there is little incentive to take action or persevere in difficult situations. The core belief of SE is that people can make a difference by actions they take.

Studies have shown that it is challenging for K-12 students to learn programming and to obtain CT skills, because programming and CT involve higher-order thinking skills such as algorithmic thinking, creative thinking, and problem solving (Kalelioğlu, 2015). As such, helping young learners to program with high SE, that is, enjoyment, motivation and confidence, is vital in cultivating their CT (Tsai, Wang & Hsu, 2019). Studies have shown statistically significant correlations between CT performance and programming SE (Román-González, Pérez-González, Moreno-León & Robles, 2018). For instance, Durak, Yilmaz & Yilmaz (2019) found that students with high performance in programming have both high SE levels and high CT skills.

3. Methodology

The purpose of this study was to examine the effectiveness of PPP on elementary school students' CT skills and programming SE. The goal was to improve elementary students' CT skills and programming SE. We therefore posed the following research questions (RQ):

RQ1: Based on their scores in Dr. Scratch, what are the CT differences between the students who did their programming assignments in pairs (Experimental Group – EG) compared to the students who did the assignments on their own (Control Group – CG)?

RQ2: Based on their SE reports, what are the SE differences between the students who did their programming assignments in pairs (EG) compared to the students who did the assignments on their own (CG)?

RQ3: What is the relationship between CT and SE for the EG students?

A mixed methods approach including surveys, instruments, and interviews was used to answer the above questions and to better understand the perspectives of these 4th grade students. Based on a 17-week experimental study of the EG and CG, we conducted pre- and post-assessments of CT skills and SE in order to find out whether PPP could improve the elementary students' CT skills and SE. We also conducted interviews with volunteer students and their teacher to explore the collaboration between the student pairs and to understand why and how PPP might affect the improvement of the students' CT skills and SE.

3.1 Research Context

The study was conducted at an elementary school in China during the Spring 2019. Four 4th grade classes with a total of 171 students participated in the study. The students were taking the course entitled "Computational Thinking with Scratch" taught by the same teacher. There were 17 class periods in total, including 16 instructional weeks and one evaluation week. There was one class per week, and each class lasted 40 minutes. Students had not taken any programming classes before. All the students had similar backgrounds and experiences in CT and programming. Two of the classes (84 students) were randomly assigned to the EG (44 boys and 40 girls), while the other two (87 students) were assigned to the CG (45 boys and 42 girls). The students in EG were paired up for their programming assignments while the students in CG did the programming by themselves. Based on prior research findings about gender differences (Zhong, Wang & Chen, 2016), the students in the EG were paired up by gender (11 boy-boy pairs, 9 girl-girl pairs, and 22 boy-girl pairs). The teacher of the four classes had 10 years of teaching experience and was considered as one of the best teachers at the school. During his teaching career, he had gathered experience teaching programming with multiple methods. He designed the instructional plan for this course and received critical feedback from two other experienced CT teachers at the local school. The detailed instructional plan of the course is shown in the Appendix A.

3.2 Instruments

In the study, we employed a mixed methods approach, which includes pre- and post-CT skill tests done by Dr. Scratch, pre- and post-programming SE surveys, and interviews at end of the course (see Figure 1).

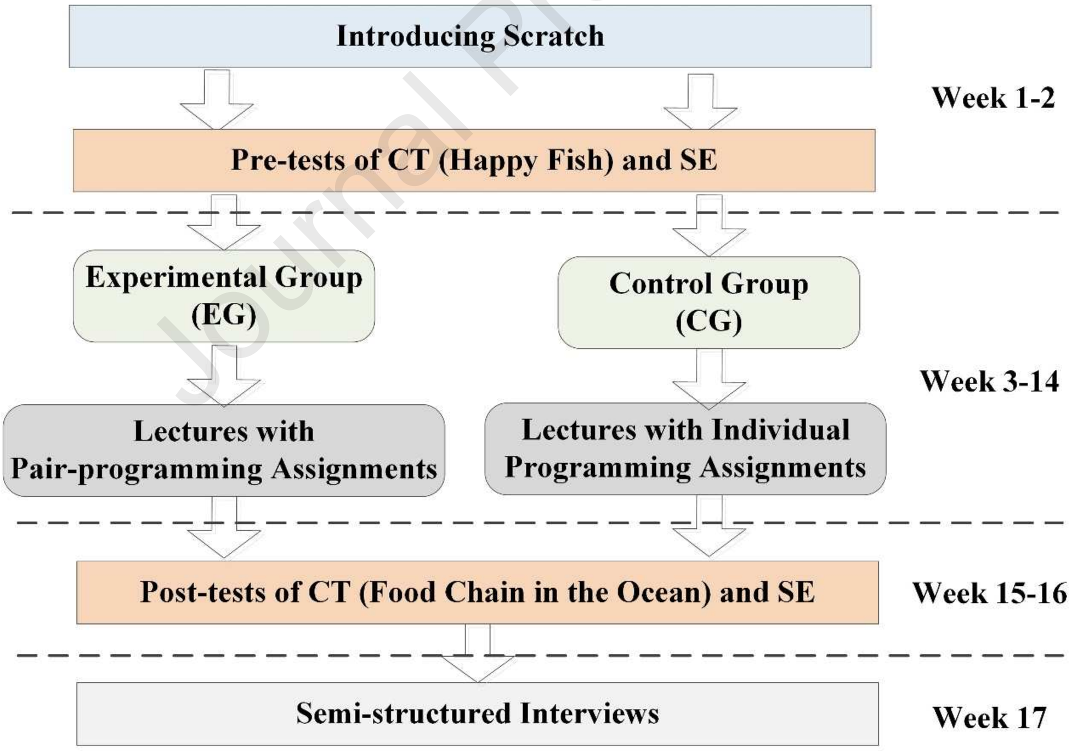
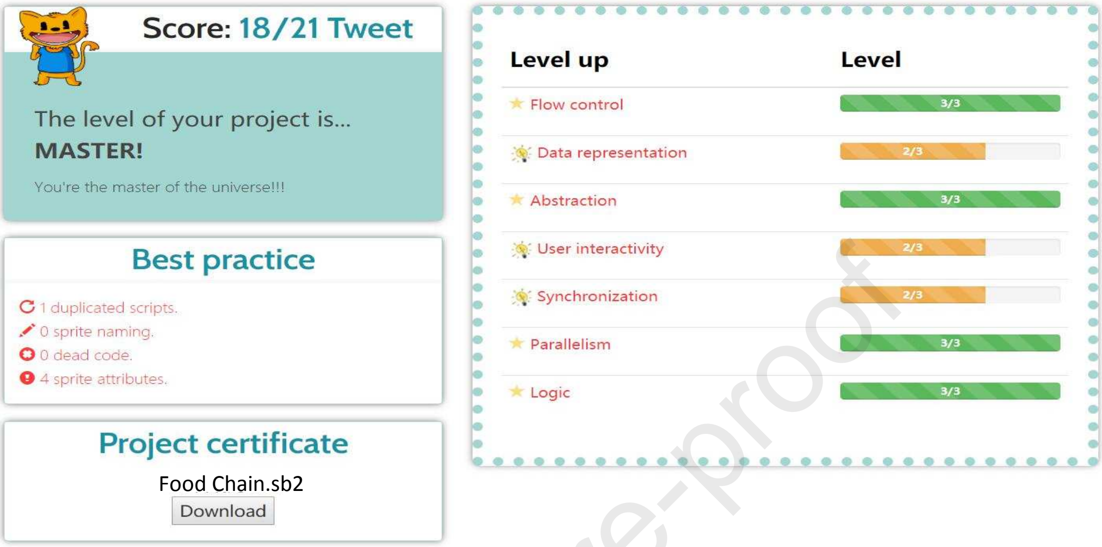

Introducing Scratch
Week 1-2
Pre-tests of CT (Happy Fish) and SE
Experimental Group (EG)
Control Group (CG)
Week 3-14
Lectures with Pair-programming Assignments
Lectures with Individual Programming Assignments
Post-tests of CT (Food Chain in the Ocean) and SE
Week 15-16
Semi-structured Interviews
Week 17

Figure 1. The Experimental Procedure

The students' CT skills were assessed through Dr. Scratch, an open-source CT assessment tool that automatically analyzes the programming projects submitted by students. This tool also provides feedback for improving students' programming and CT skills (Moreno-León & Robles, 2015). Due to the wide acceptance of Dr. Scratch as the assessment tool for Scratch programming projects (Moreno-León & Robles, 2015), the pre- and post-assessment scores generated by Dr. Scratch were recorded as the evidence of students' CT skills (those two scores range from 0 points to 21). The pre-tests were conducted during Week 2, using the scores of the students' first projects as the baseline. Specifically, the students completed their first assignments "Happy Fish" during Week 2 (see Figure 1 and Appendix A). The teacher then uploaded the students' projects to Dr. Scratch. These scores were used as a baseline for the students' prior knowledge of programming and CT skills. The post-tests were conducted during Week 16, using the scores of the students' final projects; that is, the students completed their last assignment "Food Chain in the Ocean" during Week 16. The teacher then uploaded the students' final projects to Dr. Scratch again. The students' scores of their final projects were used as post-test assessments of their CT skills. The goal was to see how well the students learned during the course of 16 weeks and to see if there were differences between students in the EG and CG.

The Scratch project entitled "Food Chain in the Ocean" created by student was shown in Appendix B. Figure 2 below shows how this "Food Chain in the Ocean" project was assessed by Dr. Scratch. Every CT project uploaded to Dr. Scratch was scored along seven dimensions: the flow control, data representation, abstraction, user interactivity, synchronization, parallelism, and logic. Each dimension was scored between 0-3 (0 being the lowest and 3 being the highest score). The score of a project was the sum of the scores of these seven dimensions. As shown in the example below (Figure 2), this specific project was scored 18 out of 21 points by Dr. Scratch. This shows that the student who created this project was at the master level of CT skills.



Score: 18/21 Tweet

The level of your project is...
MASTER!

You're the master of the universe!!!

Best practice

1 duplicated scripts.
0 sprite naming.
0 dead code.
4 sprite attributes.

Project certificate

Food Chain.sb2

Download

Level up	Level
★ Flow control	3/3
Data representation	2/3
★ Abstraction	3/3
User interactivity	2/3
Synchronization	2/3
★ Parallelism	3/3
★ Logic	3/3

Figure 2. An Example of Students' CT Projects Assessed by Dr. Scratch

To assess the students' SE levels in the current study, we adopted Kong's programming SE scale for senior primary school students (Kong, 2017). Kong (2017)'s survey was used to assess students' perceptions and judgments of their abilities to apply their programming knowledge and skills to solve computational problems. The survey consisted of 15 items, including 7 items on programming knowledge and 8 items on programming skills. With a five-point Likert Scale (1 being the lowest, 5 being the highest, points ranging from 15-75), the Cronbach's alpha value of the questionnaire was 0.92 (Kong, 2017), and the Cronbach's alpha of the two subscales was 0.88 and 0.93, respectively. This indicated that there was an acceptable internal consistency (Cortina, 1993). The survey was tested as suitable and understandable by elementary school students (Kong, 2017).

After the course, we interviewed the course instructor and the students in the EG. The purpose of the interviews was to gain a better understanding of the perspectives and experiences of the paired students and their CT skills and SE levels. We used the purposeful sampling method to select the interested participants for interviews. We wanted to determine if there might be different experiences by the participants that complement their quantitative results (Miles, Huberman & Saldana, 2013). The sample interview questions included among others: What was the most challenging project? How did you overcome the challenge? Did you work with your partner, and if so, could you tell us about it? What did you learn the most in working with your partner? What do you think is an ideal collaboration for you? Based on the quantitative results of CT and SE scores, we put the students into four profile groups: a group with high CT and high SE scores, a group with high CT and low SE scores, a group with low CT and high SE scores, and a group with low CT and low SE scores (please see Figure 5). In order to have students in each group represented (Diefenbach, 2009), we purposefully selected four volunteer students from each group for the interview (in total 16 students). All the interviews were audio-recorded and transcribed into texts for analysis. Two researchers and the teacher were involved in the data coding process. Given the fact that the interviewees were quite young (4th graders), the researchers checked and confirmed the meanings of words, phrases, and sentences used by the students with the teacher throughout the coding process. The researchers started with initial coding, constantly comparing notes, and checking with each other to ensure consistency of the coding. After the themes emerged, the researchers started to focus on the themes that were directly related to the students' descriptions of their programming skills, their methods of solving the problems encountered in the programming process, their programming experiences with their paired partners, their attitudes towards programming, and their attitudes towards the paired collaborative programming.

3.3 Experiment Procedure

Figure 1 shows the experimental procedure. During the first two weeks, the students took the CT and SE pre-tests (Kong, 2017). The students' scores of their first assignment, which was entitled "Happy Fish", were used as their pre-test CT scores. Afterwards, the students were divided into EG and CG. The teacher introduced the PPP approach to students in the EG, and helped the students become familiar with working with their partners as pairs for their programming assignments in the class. The students in the CG received the same lectures but were asked to complete their assignments in the class on their own. In week 15 and week 16, all students worked on their last programming project entitled "Food Chain in the Ocean". The students' scores of this project were used as the post-test CT scores. During the final week (week 17), all the students took the post-SE survey. Table 1 below shows the schedule of one of the "Computational Thinking with Scratch" classes for EG students, which lasted approximately 40 minutes.

Table 1. A Sample Lesson of an EG Class

Class session	Time	Details
Teacher presentation	10 minutes	The teacher presented the lecture.
Project ideas	5 minutes	Students were paired up to discuss their project ideas and the paired students proposed their ideas together.
Programming in pairs	20 minutes	Students worked in pairs on their project ideas/assignments, including discussing ideas, collecting materials, and helping each other to use Scratch.
Students' project presentations	5 minutes	All students presented their projects, explaining their ideas and design process.

Table 2 shows the schedule of one of the "Computational Thinking with Scratch" classes as an example of the CG students programming by themselves, which also lasted approximately 40 minutes.

Table 2. A Sample Lesson of a CG Class

Class session	Time	Details

Teacher presentation	10 minutes	The teacher presented the lecture.
Project ideas	5 minutes	Each student came up with their own project idea and proposed their individual idea.
Programming on their own	20 minutes	Each student worked on their own project. When they had questions, they asked the teacher.
Students' project presentations	5 minutes	All students presented their projects, explaining their ideas and programming process.

In both the EG and CG, every student had their own computer, but the paired students were asked to discuss their projects and to help each other solve problems. Each student submitted their own work. In the CG, every student worked on their own project. When they had questions, they were told to figure it out themselves or ask the teacher.

3.4 Data Analysis

In this study, the IBM SPSS was used to compare the CT skills and the programming SE of the students in the EG vs. CG. The collected data were examined first using descriptive statistics to explore the group means, numbers, and standard deviations. An analysis of covariance (ANCOVA) was conducted to examine the effects of using the PPP method on students' CT skills and programming SE. A bivariate linear correlation analysis was carried out to examine the co-relation between students' CT and SE.

Subsequently, the data of the interview was coded to discover the elementary students' perceptions and experiences. Two coders read all the interview transcripts and discussed coding themes and quotes with the help of the teacher. The first three authors of this paper discussed and reached an agreement as regards the themes and quotes reported in this study.

4. Results

The purpose of the study was to explore the effectiveness of PPP for improving the elementary students' CT skills and SE. We report our findings below.

4.1 Computational Thinking Skills

An analysis of covariance (ANCOVA) was employed using the students' pre-test scores as a covariate. The assumption of homogeneity of regression showed a common regression for a one-way ANCOVA (F = 0.76, p > 0.05), indicating that ANCOVA can be used to interpret the relationships between students' pre-test and post-test on their CT skills. Figure 3 shows the pre-test and post-test of CT between the EG and CG. Table 3 shows the scores of CT skills for the two groups generated by Dr. Scratch after the experiments. According to the results, the CT skills of the EG was significantly higher than those of the CG (F (1, 168) = 10.12, p < 0.01). The adjusted mean and standard error were 14.06 and 0.36 for the EG, and 12.44 and 0.36 for the CG. This provides a partial answer to RQ1, such that the students in the EG significantly improved their CT skills.

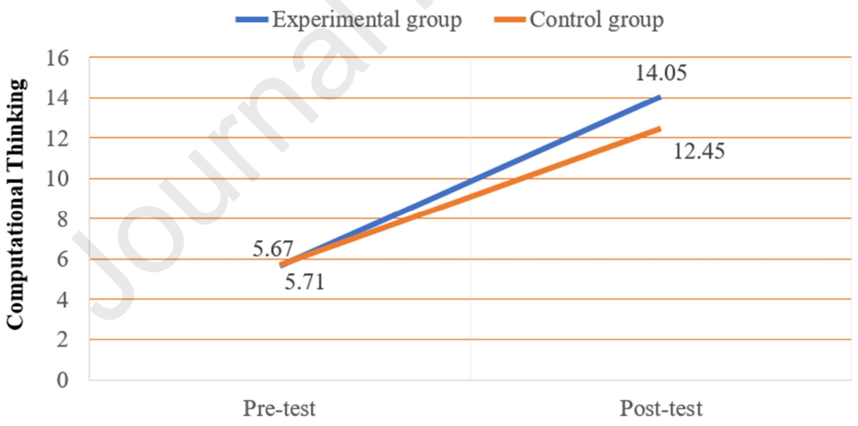


Figure 3. Pre-test and Post-test of CT Scores of the EG and CG students

Table 3. The One-way ANCOVA of Students' CT Skills in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG	84	14.05	3.43	14.06	0.36	10.12**
CG	87	12.45	3.25	12.44	0.36	

**$p < 0.01$

First we report the results for the boys and then those for the girls. Table 4 below shows the scores of boys' CT skills for the two groups after the experiment.

Table 4. The One-way ANCOVA of the Boys' and girls' CT Skills in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG (boys)	50	14.12	3.38	14.12	0.47	6.19*
CG (boys)	50	12.46	3.30	12.46	0.47	
EG (boy-boy pairs)	28	14.57	3.54	14.74	0.63	
EG (boy-girl pairs)	22	13.55	3.14	13.32	0.72	
CG (boys)	50	12.46	3.30	12.46	0.47	
EG (girls)	34	13.94	3.56	13.97	0.58	3.72
CG (girls)	37	12.43	3.24	12.41	0.56	
EG (girl-girl pairs)	12	12.92	3.83	13.10	1.01	
EG (boy-girl pairs)	22	14.50	3.36	14.43	0.73	
CG (girls)	37	12.43	3.24	12.41	0.56	

$^*p < 0.05$

According to the results, the CT skills of the boys in the EG were significantly higher than those in the CG (F (1, 97) = 6.19, p < 0.05). The adjusted mean and standard error were 14.12 and 0.47 for the boys in the EG, and 12.46 and 0.47 for the CG. This result shows that PPP significantly improved the boys' computational skills compared to the boys in the CG. The CT skills of students in the boy-boy EG were also significantly higher than those in the CG (p < 0.01). The adjusted mean and standard error were 14.74 and 0.63 for the students in the boy-boy EG, and 12.46 and 0.47 for the CG. However, the CT skills of the boys in the boy-girl EG and CG showed no significant difference (p > 0.05). The adjusted mean and standard error were 13.32 and 0.72 for the boy in the boy-girl EG, and 12.46 and 0.47 for the CG. This result indicates that students' CT skills in the boy-boy EG were significantly improved by the PPP approach.

The CT skills of the girls in the EG and CG showed no significant difference (F (1, 68) = 3.72, p > 0.05). The adjusted mean and standard error were 13.97 and 0.58 for the girls in the EG, and 12.41 and 0.56 for the CG. This result indicates that the girls' CT skills were not significantly improved by the PPP approach. The CT skills of the girls in boy-girl EG were significantly higher than those in the CG ($p < 0.05$). The adjusted mean and standard error were 14.43 and 0.73 for the girls in the boy-girl EG, and 12.41 and 0.56 for the CG. However, the CT skills of the girls in the girl-girl EG and CG showed no significant difference ($p > 0.05$). The adjusted mean and standard error were 13.10 and 1.01 for the girls in the girl-girl EG, and 12.41 and 0.56 for the CG. This result indicates that girls' CT skills in the boy-girl EG were significantly improved by the PPP approach.

In summary, the students in the EG as a whole improved their CT skills more significantly than the students in the CG. However, there were gender differences between the EG involving boys and girls depending on how they had been paired up. In general, compared to the boys in the CG, the boys in the EG improved their CT skills significantly when they worked in pairs, be it in boy-boy and boy-girl pairs.

However, compared to the girls in the CG, the girls in the EG showed a varied picture of their CT skills depending on how they had been paired up. The girls in the boy-girl pairs improved their CT skills significantly than those in the CG. Yet, compared to girls in the CG, the girls in the PPP DID NOT improve their CT skills significantly.

4.2 Programming Self-efficacy

Levene's test of determining the homogeneity of regression was not violated ($F = 0.75$, $p > 0.05$), indicating a common regression for a one-way ANCOVA. Figure 4 shows the pre-test and post-test of SE scores of the EG and CG students. Table 5 shows the results of students' programming SE for the two groups. According to the one-way ANCOVA results, the programming SE of the EG was significantly higher than that of the CG ($F(1, 168) = 8.37$, $p <$

0.01). The adjusted mean and standard error were 61.79 and 1.54 for the EG, and 54.93 and 1.51 for the CG. This result answers part of RQ2, namely that the PPP approach significantly improved students' programming SE.

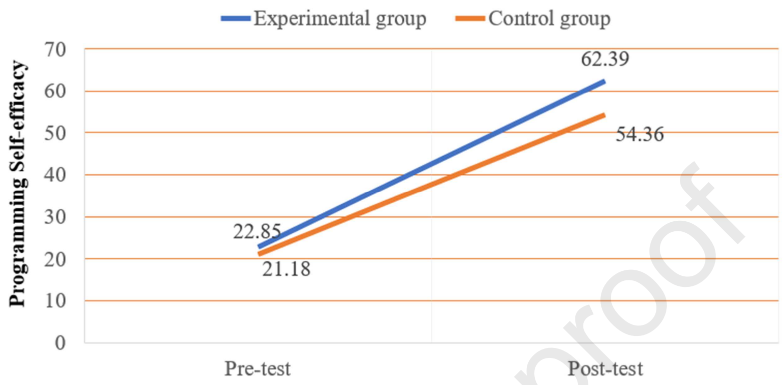


Figure 4. Pre-test and Post-test of SE Scores of the EG and CG Students

Table 5. The One-way ANCOVA of Students' Programming SE in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG	84	62.39	10.89	61.79	1.54	8.37^{**}
CG	87	54.36	13.95	54.93	1.51	

$^{**}p < 0.01$

Table 6 below shows the results of the boys' programming SE for in the EG and CG.

Table 6. The One-way ANCOVA of the Boys' and girls' Programming SE in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG (boys)	50	63.20	9.96	63.19	1.99	7.64^{**}
CG (boys)	50	54.64	14.64	54.65	1.99	
EG (boy-boy pairs)	28	62.89	9.35	62.87	2.59	
EG (boy-girl pairs)	22	63.59	10.90	63.58	2.81	
CG (boys)	50	54.64	14.64	54.65	1.99	
EG (girls)	34	61.21	12.18	59.70	2.46	1.35
CG (girls)	37	53.97	13.16	55.35	2.34	
EG (girl-girl pairs)	12	59.42	12.31	56.97	4.04	
EG (boy-girl pairs)	22	62.18	12.29	60.91	2.84	
CG (girls)	37	53.97	13.16	55.35	2.34	

$^{**}p < 0.01$

According to the one-way ANCOVA results, the boys' programming SE of the EG was significantly higher than that of the CG (F(1, 97) = 7.64, p < 0.01). The adjusted mean and standard error were 63.19 and 1.99 for the EG, and 54.65 and 1.99 for the CG. This result shows that PPP significantly improved the boys' programming SE. The programming SE of the students in the boy-boy EG was significantly higher than that of the CG (p < 0.05). The adjusted mean and standard error were 62.87 and 2.59 for the students in the boy-boy EG, and 54.65 and 1.99 for the CG. Also, the programming SE of the boys in the boy-girl EG and CG showed a significant difference (p < 0.05). The adjusted mean and standard error were 63.58 and 2.81 for the boys in the boy-girl EG, and 54.65 and 1.99 for the CG. This result indicates that the boys' programming SE in the boy-boy EG and the boy-girl EG was significantly improved by the PPP approach.

As for girls' programming SE, there was no significant difference of between the EG and the CG (F(1, 68) = 1.35, p > 0.05). The adjusted mean and standard error were 59.70 and 2.46 for the EG, and 55.35 and 2.34 for the CG. This result indicates that the girls' programming SE was not significantly improved by the PPP approach. The programming SE of the students in the girl-girl EG and the CG showed no significant difference (p > 0.05). The adjusted mean and standard error were 56.97 and 4.04 for the students in the girl-girl EG, and 55.35 and 2.34 for the CG. Also, the programming SE of the girls in the boy-girl EG and the CG showed no significant difference (p > 0.05). The adjusted mean and standard error were 60.91 and 2.84 for the girls in the boy-girl EG, and 55.35 and 2.34 for the CG. This result indicates that the girls' programming SE in the girl-girl EG and boy-girl EG was not significantly improved by the PPP approach.

In summary, after taking the programming course, the students in the EG improved their SE more significantly than the students in the CG. However, there were gender differences between the EG involving boys and girls. In general, compared to the boys in the CG, the boys in the EG improved their SE significantly more (Table 6), regardless of whether or not they had been paired up with other boys or with girls. In contrast, the girls in the EG did not improve their SE significantly more than the girls in the CG. This was true for the girls in general, and is true whether or not the girls had been paired up with other girls or with boys.

4.3 Relationships between Computational Thinking and Self-efficacy

4.3.1 Correlation between Computational Thinking and Self-efficacy

An analysis of bivariate linear correlation was conducted to assess the relationship between students' CT and programming SE. As shown in Table 7, in the pre-test, there was no a correlation between CT and programming SE in the EG (r = 0.04, n = 84); nor was there correlation between CT and programming SE in the CG (r = 0.01, n = 87). After the experiment, there was a low correlation between CT and programming SE in the post-test scores of the EG (r = 0.24, n = 87, p < 0.05). There was also a low correlation between CT and programming SE in the post-test scores in the CG (r = 0.19, n = 87).

Table 7. The Correlation between the Students' CT and Programming SE

Group	CT and SE Tests	N	Pearson Correlation coefficient (r)
EG	Pre-tests	84	0.07
CG	Pre-tests	87	0.01
EG	Post-tests	84	0.24^*
CG	Post-tests	87	0.19

$^*p < 0.05$

Overall, the correlation between CT and programming SE became slightly stronger in students of both the EG and CG. Yet, the correlation was low.

4.3.2 Interviews of the Students

Figure 5 below displays the post-test CT and SE scores of the 84 students in the EG.

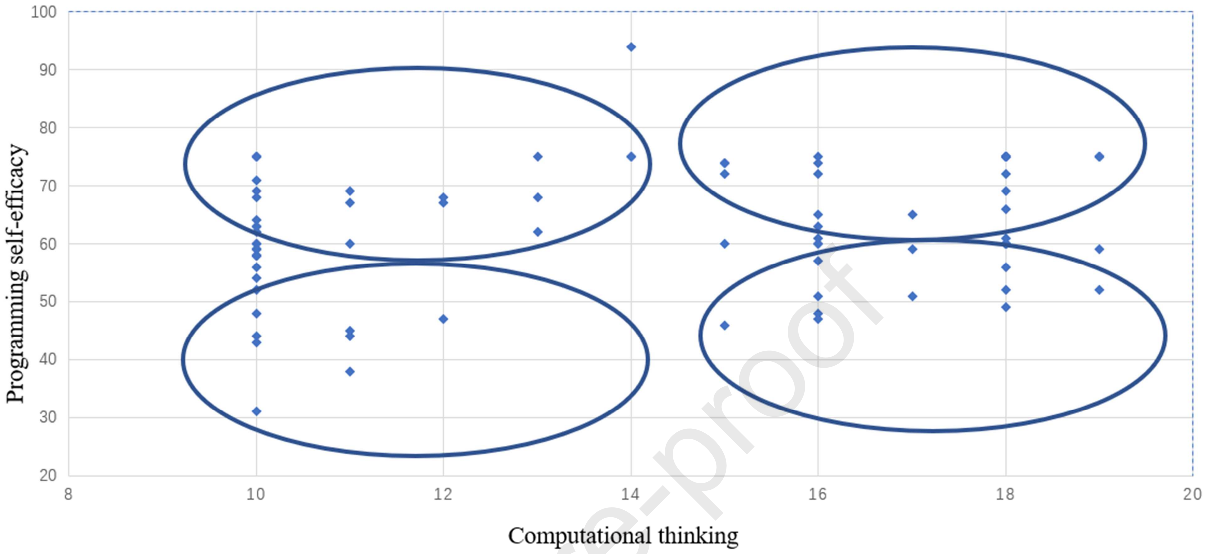


Figure 5. EG Students' Post-CT and Programming SE Scores

Although not exactly proportionally distributed, we could categorize the students into four profile groups based on their CT and SE scores. Specifically, some students had high SE but low CT scores (upper left); some had low SE and low CT scores (lower left); some had low SE but high CT scores (lower right); and some had high CT and high SE scores (upper right). To better understand the students' PPP experience and the potential connections between PPP, CE, and SE (RQ3), we purposefully selected students from each of the four profiles for interviews (Miles, Huberman & Saldana, 2013). A total of 16 volunteer students, four representing each profile group, were selected for the one-on-one interviews. These included four students with both high CT and high SE scores; four with high CT but low SE scores; four with low CT but high SE scores; and, four with low CT and low SE scores.

The sample interview questions included: What was the most challenging project? How did you overcome the challenge? Did you work with your partner, and if so, could you tell us about it? What did you learn the most in working with your partner? What do you think is an ideal collaboration for you? All interviews were recorded and transcribed verbatim. A qualitative thematic analysis (Miles, Huberman & Saldana, 2013) was conducted by two coders/researchers and double-checked with the teacher given the fact that the interviewees were young (4th graders). Notes were then gathered and compared. After themes emerged, the two coders, the teacher, and another senior researcher started to focus on the themes that were directly related to the students' descriptions of their programming skills, their methods of solving the problems encountered during the programming process, their programming experiences with their paired partners, their attitudes towards programming, and their attitudes towards the paired collaborative programming. Below, we will discuss the themes from each group as well as provide a representative quote from an interviewee to illustrate the themes.

For the students with high CT skills and high SE, we discovered that 1) these students could describe their programming work in clear terms; 2) the students considered programming not only as a learning process, but also as an enjoyable experience; 3) the students enjoyed programming with their partners, as they could easily brainstorm problems they encountered; 4) the students showed strong desires to share their work with others, with improved confidence in programing; and 5) the students expressed a strong desire to challenge themselves with more complex programming skills within and beyond the requirements of the programming course. Two of the interviewed students indicated that their improved programming skills made learning other subjects more easy because of their improved reasoning skills.

The following quote is from a student named Xu, who scored high in both CT and SE:

> "When I encountered a problem in designing my project, I would think about it myself first. I would then talk with my partner. During the semester, I would talk with my partner when I couldn't solve a problem. When neither of us could solve the problem, we would talk about it in more depth. Often, we could solve the problem together. This is

> also the collaborative process I had wished for. I didn't mind sharing my work. I was very happy when I could solve the problem by myself. Yet, my partner was also very helpful. The most exciting part was when I could solve more difficult and challenging problems. I continued to increase the difficulty levels of the projects myself. Solving more challenging problems made the process much more fun. After this experience, I can apply my thinking skills to other subjects; the code I wrote is getting clearer." (Translated from Chinese to English)

It is clear that this high CT-high SE student was confident in his own ability to solve problems. Xu expressed that his reasoning, logic and problem solving greatly improved, and learning programming skills helped him with learning other subjects. He could transfer the improved CT to other subjects. Xu enjoyed solving difficult problems, which showed his high level of SE. Meanwhile, he also appreciated the collaboration and help from his partner. He particularly highlighted that the best way to solve a difficult problem was to work on the problem together with his partner. Xu not only acquired the skills of programming, but also enjoyed programming. With improved confidence in programing, Xu showed a strong desire to share his work with others, and to challenge himself with more complex programming skills.

For those with high CT but low SE scores, the students 1) focused their comments heavily on the steps they went through to carry out their ideas and programming processes in order to make the program work; 2) appeared to be hesitant when asked if they wished to share their program work with others; 3) did not mention a sense of achievement (as the students with high CT and high SE did) after completing their programming work; and 4) enjoyed programing with their partners, solving the problems together. The following quote is from a high CT-low SE student called Wang:

> "When I had a problem or did not know what to do, I would ask my partner. If my partner did not know the answer either, we would talk, think together, and try to figure out how to do it together. I feel that there were a lot of difficult problems that I did not

> know how to solve; yet, with my partner, we were able to figure out a lot more solutions. Sometimes my partner and I couldn't figure it out together either. We would then ask the other classmates. We learned a lot and we learned to better communicate with each other. There were some problems too. For instance, sometimes we had different ideas; sometimes neither of us knew the answer. To me, the ideal collaboration is that both people have their own ideas, and that both people can reach their goals. My partner and other classmates also helped a lot. During the semester, I coded some small programs and projects based on my personal life. I was very excited when I successfully programmed a project by myself and when I was able to complete the projects as I had hoped to. My biggest achievement was that I acquired a lot of new knowledge and learned new codes, e.g., how to write code to insert background music. I would like to add more sophisticated coding and make my work more complex, rich and beautiful based on previous work."

This high CT-low SE student indicated that she and her partner were able to solve some problems. When they could not solve the problems, they could ask other people to help them with the problems. Wang had enough confidence to know where to look for solutions. However, she did not have as much confidence as the previous student in expressing her level of expertise or her ability to solve problems. She always expressed that there was room for improvement.

For the students who scored low on CT but high on SE, they revealed the following characteristics: 1) the students could not describe the main ideas of their programming work with organized and clear language; 2) the students' attitudes toward programming appeared to be positive; 3) the students focused their comments heavily on their collaboration experience and on how much they enjoyed collaborating with their partners. A student named Sun received a low CT but high SE scores. Here is what he had to say:

> "The biggest difficulty for me was to add the music score to my project. I could not make the music play or the music did not come out smoothly. I would wait for a few seconds before repeating the music. In the process of creating my work, I would work with my partner. For instance, there was a procedure I could not figure out. I asked my partner quietly. I asked him what he thought. Then I told him my thinking. Then I would

> try it out on the computer and see if the solution would work. There was no way I could find so many solutions by myself; working with my partner together, we could find many solutions. Sometimes we couldn't agree. For instance, he would take steps in one direction, and I would suggest taking steps in a different direction. We then would test them on the computer and see which one was better. I liked the smooth collaborations with my partners; this way, we could get work done quickly. I shared my work with the others. For instance, I created some games at home. I brought the games to the class and showed them to my classmates. I also showed my classwork at home. I was very happy and proud. The one who helped me the most was my partner. Whenever I had difficulty programming, I would talk with my partner. Then I would be able to figure out how to program. I will communicate more with my partner in the future and do a better job with my programming projects."

Sun was not as eloquent in his language about his CT. He indicated that he had good self-confidence and thinking in solving problems. It also showed that his approach to solve problems was good. He was good at communicating with his partner and good at sharing with his classmates at school. Sun was pleased with the process. Although Sun might not be able to conduct as many programming tasks compared to students with high CT, he enjoyed the collaborative process and working with his partner, which helped boost his confidence level toward programming.

For the students who received both low CT and low SE scores, they showed the following traits: 1) none of the students we interviewed shared their programming processes willingly; and when they were asked, they could not clearly elaborate on what they accomplished in the programming tasks; 2) the students could not articulate how their collaboration with their partners helped their programming, although they were positive about PPP and their collaborative experiences. A student called Liu received low CT and low SE scores. Here's what she had to say:

> "When encountering a problem, I would try to solve it with my partner in the class. If I could not solve the problem in the class, I would go home and try to use WeChat to

> communicate with my partner. During the collaborative process, I learned that I could always ask my partner if I did not know what to do. My partner helped me a lot. I think that it is important for partners to help each other and it is important to try our best to do the best job possible during the collaborative learning process. During the semester, I did not share my work with others, but in the future I will learn to share my work. What was most exciting to me was that my partner and I could do a good job together and that we learned a lot from helping each other. I think I will be able to do a better job next time."

During the interview, Liu focused more on the unresolved problems. This might imply that Liu had quite some problems that she was not able to solve (consonant with her low CT). Liu highlighted her positive collaborative experiences with her partner, although in discussing her collaboration, Liu did not mention any specific CT programming steps. Instead, she emphasized the importance of collaboration itself. The possible reasons behind this observation could be: 1) There was not enough improvement in her CT for Liu to comment on (low CT); 2) she did not enjoy programming (low SE); 3) compared to learning programming and the sense of confidence and accomplishment, Liu enjoyed the collaborative learning format itself more than the actual content. If this is the case, then it is important for educators to make a note of that. This may indicate another important aspect of collaboration, namely PPP as discussed in this paper. In this particular context, PPP helped the student to continue her interest in programming because of her interest in working and collaborating with others, although it did not immediately help the student improve the CT or SE during the given course. Table 8 below presents a summary of the main characteristics of the students based on their CT skills and SE.

Table 8. Main Characteristics of the Four Groups of Students

CT	SE	N	Main Characteristics
High	High	4	Confident in solving CT problems; using proper CT language; interested in solving more challenging problems; enjoyed programming collaboratively.
High	Low	4	Confident in ways to solve problems; yet emphasizing continuous improvements in programming; hesitant to share their programming work with others; enjoyed programming with their partners and solving the problems together.
Low	High	4	Lack of CT domain-specific language; yet having no problem looking for solutions to the problems; enjoying collaborative learning experiences.
Low	Low	4	Avoiding discussions of programing processes; discussing the challenges of solving problems; highlighting the enjoyment of working with others.

In summary, regardless of students' SE and CT skills, it is clear that the students all appreciated working with their peers in the PPP assignments. This result does not only align with the result of Kong, Chiu and Lai (2018) that positive attitude toward collaboration can bring students enjoyable experience in learning content in programming, but it also shows that a positive attitude toward collaboration may help encourage students to continue to try programming. The sustained effort and time may potentially be the key to help students develop an interest and competence in CT.

4.3.3 Interview with the Teacher

In order to gain some in-depth insights into the students' CT skills and into the gender differences as regards PPP, we interviewed the teacher of the four classes. The interview questions mainly focused on the differences between the students' programming skills, programming attitude and gender differences between the EG and CG. The interview with the teacher was also recorded, transcribed verbatim, and underwent qualitative analysis by two researchers. The following is a quote from the teacher:

> "Students' programming in pairs showed greater skills and attitude towards programming than students' programming on their own. Students in pairs showed agile thinking in the class. The quality of their work was higher. Pair work facilitated the development of the communication and problem solving skills for both boys and girls. Additionally, there was only one teacher available for over 30 students in the class. This led to two problems: the teacher, me, could not always spend time paying attention to every student who needed my

> help; there were experience and language differences between the students and me. I was not always able to solve the problems in the way they could easily understand. PPP helped the students who had similar experiences and interests to better understand each other. This made programming more accessible to them. It is worth noting that some student pairs worked better than others – students paired up with friends were more excited than students paired up with less familiar peers. In addition, generally speaking the boys showed stronger logic thinking skills than the girls, but the girls showed superior oral communication skills. Boy-girl pairs showed that they had complementary skills, and that the quality of their programming work seemed to be much better than those who programmed on their own. In solo programming classes, I was the only person that students could ask for help. Boys seemed to have a stronger desire to solve the problems they encountered, with or without the teacher's help. The girls seemed to give up more quickly than the boys, especially when they could not get the teacher's help."

The teacher's observation revealed the differences of the class dynamics between students programming in pairs and alone, as well as the gender differences. In a typical elementary classroom (especially in a large-size class) in China, students are not allowed to talk to each other without the teacher's permission. PPP compensated for the shortage of the teacher's availability in such large-size classes, and promoted the students' collaboration by allowing them to contribute their strengths and learn from each other. Yet, the familiarity between two students within a pair also seemed to affect the dynamics of the collaborative work. The teacher's insights also reflected the differences of persistence in problem solving between boys and girls. Girls demonstrated stronger communication skills and a stronger desire to develop CT skills during the collaborative work; while boys showed a willingness to solve the problems they encountered. These observations supported the findings of Bruckman, Jensen & DeBonte (2002) in that girls spent significantly more time than boys in communicating with others during programming, and boys spent longer time programming. The teacher's insights also implied that more support and attention should be given to girls in developing programming and CT skills.

5. Discussion

This present study showed that PPP improved the elementary students' CT skills and SE. This result confirmed the findings of Denner, Werner, Campe & Ortiz (2014) and Durak, Yilmaz & Yilmaz (2019) regarding the positive impact of PPP on middle school students, and corroborated the findings of Iskrenovic-Momcilovic (2019) that beginning students who programmed in pairs performed better than those who did programming on their own. Meanwhile, this study also extended findings from prior studies, and showed the positive impact of PPP on CT and SE for elementary school students.

The present study found no correlation between CT and SE before the experiment, and a low correlation between CT and SE after the experiment, in both the EG and CG. It is important to point out the existence of a low correlation between CT and SE, but it is also worth noting that PPP as the pedagogical treatment did not lead to a stronger correlation between CT and SE.

Another finding of this study was that, even for the students who did not seem to improve their CT or SE significantly during the PPP learning process, these students enjoyed collaborations with their partners, and that they had positive attitudes toward further development of their programming skills. Such an enjoyment is important for students' continued interest and motivation for learning to program. The positive feeling and experience may potentially help the students to stay and make continued efforts to learn rather than to give up solving computational problems. As pointed out by Duckworth, Peterson, Matthews & Kelly (2007), perseverance and passion for long-term goals are the best prediction of success. It takes prolonged interest and practice, dedicated time and a growth mindset for students to achieve their academic goals. Our study showed that PPP is an important learning experience regardless of whether or not this method may immediately help students improve their CT skills or SE.

Through PPP, students showed interest in learning and willingness to take on challenging tasks. Although in reality, the students in the PPP groups had less time working on their individual projects (due to spending time helping one another), they still had higher CT scores. This shows that PPP played an important role in helping students acquire CT skills. All this will benefit their future learning.

During the PPP collaboration process, the paired students were informed of and provided with opportunities to play different roles such as the navigator and driver. As reflected by the interviewed students, they would talk to their partner and work out the problems together when they each encountered a problem, or they would talk to the other classmates or the teacher if both of them failed to resolve the issue at hand. Such behaviors were more frequently demonstrated by students with high SE. Students enjoyed assisting or mentoring each other during the problem solving processes. This observation is consistent with the result suggested by Clarke-Midura et al. (2018) that mentoring helped students improve their SE in programming and boost their general interest in studying computer science.

With young programming learners, friendships between two partners are worth taking into account as well. The observation of the teacher that students who were paired up with their friends enjoyed more and performed better than the students who were paired up with non-friends. This is consistent with the observations reported in the study by Campe, Denner, Green & Torres (2020). This observation should remind programming teachers to take friendships or familiarities between two students into consideration when pairing up students to work together, as good communication between friends could contribute to students' interest and ability to obtain scientific knowledge (Leman et al., 2016) and could help facilitate collaborative processes (Sullivan & Wilson, 2015).

The large class size may also help explain why PPP worked well in the context of this study. The current experiment was conducted in a public elementary school in east China. Each class consisted of over 40 students, which is the typical class size in the schools there. Due to the large class size, the teacher may not be able to pay attention and spend time answering questions from all the students. Consequently, it could be more efficient and effective for the students to work with their peers in pairs and seek help from each other, than waiting for help from their teacher. However, this statement needs further empirical evidence so that a more solid connection may be established between the classroom size and the efficiency or effectiveness of PPP in K-12 programming classes.

The current study also touched on the gender differences between elementary boys and girls in the programming class. As discussed previously, the students in the EG improved their CT and SE significantly more than the students in CG did. However, the boys seemed to have benefited more in both CT and SE in this PPP setting. The boys in the EG improved their CT and SE significantly more than the boys in CG, as well as when they were paired up with girls. This was not the case for the girls. Only the girls who were paired up with boys improved their CT skills significantly more highly than the girls in the CT groups did. The girls did not improve their CT scores significantly more than the girls in the CG, or when they were paired up with other girls. The girls in the EG or paired up did not improve their SE scores significantly more than the girls in the CG did. In other words, PPP may help, but pairing up girls with girls may not help improve CT skills or SE. Further empirical studies are needed to investigate the motivation and pedagogy that would help girls to develop CT skills and programming SE.

6. Limitations and Conclusion

This study is not without limitations. First, the study was conducted at one elementary school with one classroom setting in northern China, and the experiment lasted 17 weeks. Although typical in this area, there are some unique features of the context that may be different from the classrooms in the other parts of the world. These features include the large-size classes, the limited teacher availability to working with individual students, and the fact that students are not allowed to talk to each other without getting permissions from the teacher in the classroom (Stevenson & Stigler, 1992). Future research is needed to validate the results with a more diverse population, over a longer period of time or higher frequency of classes being taught. In this study, although we had the same teacher instructing the students in both the EG and CG, other factors could have affected the results. Such factors could include but are not limited to the students' programming interests and their parents' attitudes. Second, the course was designed based on the teacher's personal experience with the students, rather than following a sequential level of coding difficulty or considering the learners' personal interests. We suggest that future studies take into considerations of the design and selection of the programming projects, as well suggested by Moreno-León, Robles, G & Román-González (2017) in that a learning path with increasing difficulty could help students mastering different types of projects when introducing the programming concepts to students. In addition, it is essential to investigate the impacts of different skill-level-pairing in paired-programming assignments. Future research is needed to look into pairing up students of different skill levels, i.e. the pairings of different genders with high-to-high, high-to-low, and low-to-low skill levels, as the gender difference and skill levels might affect students' learning experiences differently.

This current study reported the effectiveness of PPP on elementary school students' CT skills and SE for the purpose of helping elementary school students to acquire CT skills effectively. The results indicated that the PPP method improved students' CT skills and programming SE, and that there was a low correlation between CT and programming SE with PPP. The interviews helped us better understand the students' perspectives and experiences with PPP. The findings suggested that K-12 programming teachers could use the PPP approach to improve students' CT skills and programming SE. In fact, after this study, the teacher was invited to demonstrate how to use the PPP method in teaching CT skills and he was commended by the local educational agency for his innovative and high-quality programming course. Since then several neighboring schools have also adopted his method for teaching elementary school students programming and CT classes.

References

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. https://doi.org/10.1016/j.robot.2015.10.008.

Bandura, A. (2010). Self□efficacy. *The Corsini encyclopedia of psychology*, 1-3. https://doi.org/10.1002/9780470479216.corpsy0836.

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology, 38*(6), 20-23.

Bear, G. G., Chen, D., Mantz, L. S., Yang, C., Huang, X., & Shiomi, K. (2016). Differences in classroom removals and use of praise and rewards in American, Chinese, and Japanese schools. Teaching and Teacher Education, 53, 41-50.

Bell, J., & Bell, T. (2018). Integrating Computational Thinking with a Music Education Context. *Informatics in Education*, *17*(2), 151-166.

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. https://doi.org/10.1080/0020739X.2016.1254296.

Boissy, A., Windover, A. K., Bokar, D., Karafa, M., Neuendorf, K., Frankel, R. M. & Rothberg, M. B. (2016). Communication skills training for physicians improves patient satisfaction. *Journal of general internal medicine*, *31*(7), 755-761. https://doi.org/10.1007/s11606-016-3597-2.

Braught, G., Eby, L. M., & Wahls, T. (2008, March). The effects of pair-programming on individual programming skill. In ACM SIGCSE Bulletin (Vol. 40, No. 1, pp. 200-204). ACM. https://doi.org/10.1145/1352322.1352207.

Brown, P. L., Concannon, J. P., Marx, D., Donaldson, C. W., & Black, A. (2016). An Examination of Middle School Students' STEM Self-Efficacy with Relation to Interest and Perceptions of STEM. *Journal of STEM Education: Innovations & Research*, *17*(3).

Browning, S. F. (2017). Using Dr. Scratch as a Formative Feedback Tool to Assess Computational Thinking. *Theses and Dissertations*. 6659. https://scholarsarchive.byu.edu/etd/6659

Bruckman, A., Jensen, C., & DeBonte, A. (2002, January). Gender and programming achievement in a CSCL environment. In Proceedings of the conference on computer support for collaborative learning: Foundations for a CSCL community (pp. 119-127).

Buffum, P. S., Lobene, E. V., Frankosky, M. H., Boyer, K. E., Wiebe, E. N., & Lester, J. C. (2015, February). A practical guide to developing and validating computer science knowledge assessments with application to middle school. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 622-627). ACM.

Campe, S., Denner, J., Green, E., & Torres, D. (2020). Pair programming in middle school: variations in interactions and behaviors. Computer Science Education, 30(1), 22-46.

Chong, J., & Hurlbutt, T. (2007, May). The social dynamics of pair programming. In 29th International Conference on Software Engineering (ICSE'07) (pp. 354-363). IEEE.

Clarke-Midura, J., Poole, F., Pantic, K., Hamilton, M., Sun, C., & Allan, V. (2018, February). How near peer mentoring affects middle school mentees. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 664-669). ACM.

Cortina, J. M. (1993). What is coefficient alpha? An examination of theory and applications. *Journal of applied psychology*, *78*(1), 98.

Dagiene, V., & Stupuriene, G. (2016). Bebras--A Sustainable Community Building Model for the Concept Based Learning of Informatics and Computational Thinking. *Informatics in education*, *15*(1), 25-44.

D'Angelo, S., & Begel, A. (2017, May). Improving communication between pair programmers using shared gaze awareness. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 6245-6290). ACM. https://doi.org/10.1145/3025453.3025573.

Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education*, *46*(3), 277-296. https://doi.org/10.1080/15391523.2014.888272.

Diefenbach, T. (2009). Are case studies more than sophisticated storytelling?: Methodological problems of qualitative empirical research mainly based on semi-structured interviews. Quality & Quantity, 43(6), 875.

Dongo, T. A., Reed, A. H., & O'Hara, M. T. (2016). Exploring Pair Programming Benefits for MIS Majors. *Journal of Information Technology Education: Innovations in Practice, 15,* 223-239. Retrieved from http://www.informingscience.org/Publications/3625.

Duckworth, A. L., Peterson, C., Matthews, M. D., & Kelly, D. R. (2007). Grit: Perseverance and passion for long-term goals. *Journal of Personality and Social Psychology, 92*(6), 1087-1101.

Durak, H. Y. (2018). Digital story design activities used for teaching programming effect on learning of programming concepts, programming self-efficacy, and participation and analysis of student experiences. *Journal of Computer Assisted Learning*, *34*(6), 740-752. https://doi.org/10.1111/jcal.12281.

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. https://doi.org/10.30935/cet.554493.

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. https://doi.org/10.1109/TPC.2016.2516639.

Friedman, J., & Jacobson, N. (2018). Computation in practice: An inquiry into the business of computational design. In *Codify* (pp. 39-49). Routledge.

Gerlach, J. M. (1994). Is This Collaboration? *New directions for teaching and learning, 59,* 5-14.

Hanks, B. (2008). Empirical evaluation of distributed pair programming. *International Journal of Human - Computer Studies, 66*(7), 530-544. https://doi.org/10.1016/j.ijhcs.2007.10.003.

Hsia, L. H., Huang, I., & Hwang, G. J. (2016). Effects of different online peer-feedback approaches on students' performance skills, motivation and self-efficacy in a dance course. *Computers & Education, 96*, 55-71. https://doi.org/10.1016/j.compedu.2016.02.004.

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

Iskrenovic-Momcilovic, O. (2019). Pair programming with scratch. Education and Information Technologies, 24(5), 2943-2952.

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior, 52*, 200-210. https://doi.org/10.1016/j.chb.2015.05.047.

Knuth, D. E. (2014). *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional.

Kong, S.C. (2017, July). Development and validation of a programming self-efficacy scale for senior primary school learners. In S.C. Kong, J. Sheldon & K.Y. Li (Eds.)*, Conference Proceedings of International Conference on Computational Thinking Education 2017* (pp. 97-102). Hong Kong: The Education University of Hong Kong.

Kong, S.C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education*, *127*, 178-189. https://doi.org/10.1016/j.compedu.2018.08.026.

Leman, P. J., Skipper, Y., Watling, D., & Rutland, A. (2016). Conceptual change in science is facilitated through peer collaboration for boys but not for girls. Child development, 87(1), 176-183.

Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *Journal of Science Education and Technology*, *25*(6), 860-876. https://doi.org/10.1007/s10956-016-9628-2.

Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? *Computer Science Education, 21*(2), 105-134. https://doi.org/10.1080/08993408.2011.579805.

Li, Z., Plaue, C., & Kraemer, E. (2013, May). A spirit of camaraderie: The impact of pair programming on retention. In *2013 26th International Conference on Software Engineering Education and Training (CSEE&T)* (pp. 209-218). IEEE. https://doi.org/10.1109/CSEET.2013.6595252.

Liebenberg, J., Mentz, E., & Breed, B. (2012). Pair programming and secondary school girls' enjoyment of programming and the subject information technology (IT). *Computer Science Education, 22*(3), 219-236. https://doi.org/10.1080/08993408.2012.713180.

Lin, J. M., & Liu, S. (2012). An investigation into parent-child collaboration in learning computer programming. *Journal of Educational Technology & Society*, *15*(1), 162-173. Retrieved from http://www.jstor.org/stable/jeductechsoci.15.1.162.

Lockwood, J., & Mooney, A. (2017). Computational Thinking in Education: Where does it fit? A systematic literary review.

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM, 49*(8), 90-95. http://doi.acm.org/10.1145/1145287.1145293.

Miles, M. B., Huberman, A. M. and Saldana, J. (2013). Qualitative Data Analysis, A Methods Sourcebook, Third Edition, *Sage Publications, Inc.*, 408 pages

Moreno-León, J. & Robles, G. (2014, October). A*utomatic Detection of Bad Programming Habits in Scratch: A Preliminary Study.* (IEEE) Frontiers in Education Conference, 2014 IEEE.

Moreno-León, J. & Robles, G. (2015, August). Analyze your Scratch projects with Dr. Scratch and assess your computational thinking skills. In *Scratch conference* (pp. 12-15).

Moreno-León, J., Robles, G., & Román-González, M. (2016, April). Comparing computational thinking development assessment scores with software complexity metrics. *IEEE Global Engineering Education Conference, EDUCON, 10-13* (April), 1040-1045. https://doi.org/10.1109/EDUCON.2016.7474681.

Moreno-León, J., Robles, G., & Román-González, M. (2017). Towards data-driven learning paths to develop computational thinking with scratch. IEEE Transactions on Emerging Topics in Computing.

Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C., & Balik, S. (2003, February). Improving the CS1 experience with pair programming. In *ACM SIGCSE Bulletin* (Vol. 35, No.1, pp.359-362). ACM. https://doi.org/10.1145/792548.612006.

Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2018). Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children? *Computers in Human Behavior.* In Press. https://doi.org/10.1016/j.chb.2018.12.027.

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, *45*(5), 583-602. https://doi.org/10.1007/s11251-017-9421-5.

Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004, June). Self-efficacy and mental models in learning to program. In *ACM SIGCSE Bulletin* (Vol. 36, No. 3, pp. 171-175). ACM. https://doi.org/10.1145/1026487.1008042.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K. & Kafai, Y. B. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60-67. http://doi.acm.org/10.1145/1592761.1592779.

Román-González, M. (2015) Computational thinking Test: Design guidelines and content validation. In 7th annual international conference on education and new learning technologies (Barcelona: Spain).

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, *72*, 678-691. https://doi.org/10.1016/j.chb.2016.08.047.

Román-González, M., Pérez-González, J., Moreno-León, J. & Robles, G. (2018). Extending the nomological network of computational thinking with non-cognitive factors. *Computers in Human Behavior, 80*, 441-459. https://doi.org/10.1016/j.chb.2017.09.030.

Rubinstein, A., & Chor, B. (2014). Computational thinking in life science education. *PLoS computational biology*, *10*(11), e1003897. https://doi.org/10.1371/journal.pcbi.1003897.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using “Scratch” in five schools. *Computers & Education*, *97*, 129-141. https://doi.org/10.1016/j.compedu.2016.03.003.

Sarpong, K. A. M., Arthur, J. K., & Amoako, P. Y. O. (2013). Causes of failure of students in computer programming courses: The teacher-learner Perspective. *International Journal of Computer Applications*, *77*(12), 27-32.

Sfetsos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2006, June). Investigating the impact of personality types on communication and collaboration-viability in pair programming–an empirical study. In *International Conference on Extreme Programming and Agile Processes in Software Engineering* (pp. 43-52). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11774129_5.

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

Stevenson, H., & Stigler, J. (1992). Why our schools are failing and what we can learn from Japanese and Chinese education. New York: Summit.

Sullivan, F. R., & Wilson, N. C. (2015). Playful talk: Negotiating opportunities to learn in collaborative groups. Journal of the Learning Sciences, 24(1), 5-52.

Sullivan, K., Byrne, J. R., Bresnihan, N., O'Sullivan, K., & Tangney, B. (2015, October). CodePlus — Designing an after school computing programme for girls. In *2015 IEEE*

Frontiers in Education Conference (FIE) (pp. 1-5). IEEE. https://doi.org/10.1109/FIE.2015.7344113.

Tsai, M. J., Wang, C. Y., & Hsu, P. F. (2019). Developing the computer programming self-efficacy scale for computer literacy education. *Journal of Educational Computing Research*, *56*(8), 1345-1360. https://doi.org/10.1177/0735633117746747.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127-147. https://doi.org/10.1007/s10956-015-9581-5.

Williams, L., Wiebe, E., Yang, K., Ferzli, M., & Miller, C. (2002). In support of pair programming in the introductory computer science course. *Computer Science Education, 12*(3), 197-212. https://doi.org/10.1076/csed.12.3.197.8618.

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

Yang, Y. F., Lee, C. I., & Chang, C. K. (2016). Learning motivation and retention effects of pair programming in data structures courses. *Education for Information, 32*(3), 249-267. https://doi.org/10.3233/EFI-160976.

Yildiz Durak, H. (2018). Digital story design activities used for teaching programming effect on learning of programming concepts, programming self□efficacy, and participation and analysis of student experiences. *Journal of Computer Assisted Learning, 34*(6), 740-752. https://doi.org/10.1111/jcal.12281.

Ying, K. M., Pezzullo, L. G., Ahmed, M., Crompton, K., Blanchard, J., & Boyer, K. E. (2019, February). In Their Own Words: Gender Differences in Student Perceptions of Pair

Programming. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 1053-1059). ACM. https://doi.org/10.1145/3287324.3287380.

Zhong, B., Wang, Q., & Chen, J. (2016). The impact of social factors on pair programming in a primary school. *Computers in Human Behavior, 64*, 423-431. https://doi.org/10.1016/j.chb.2016.07.017.

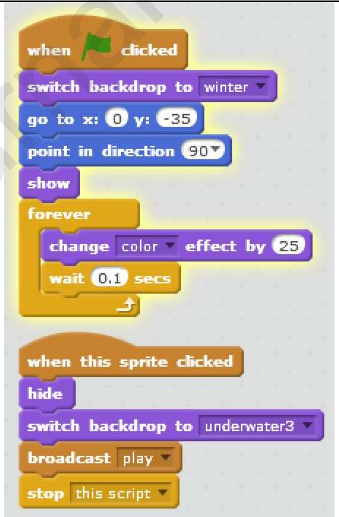
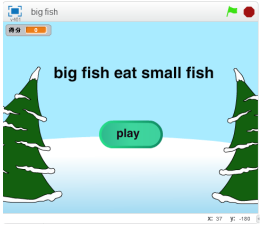
Zhong, B., Wang, Q., Chen, J., & Li, Y. (2017). Investigating the period of switching roles in pair programming in a primary school. Journal of Educational Technology & Society, 20(3), 220-233.

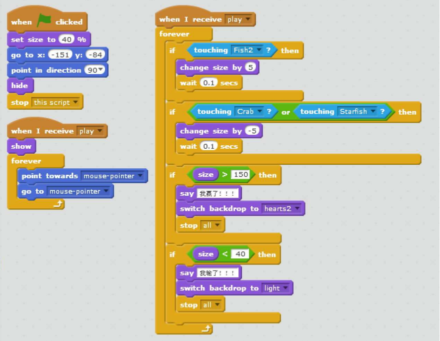
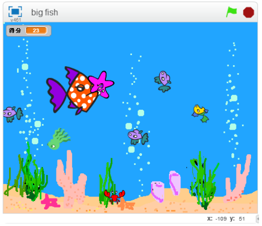
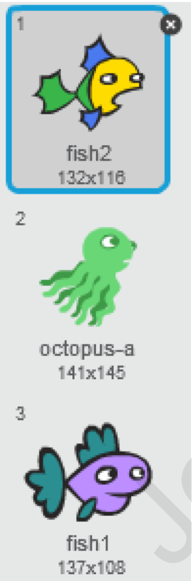
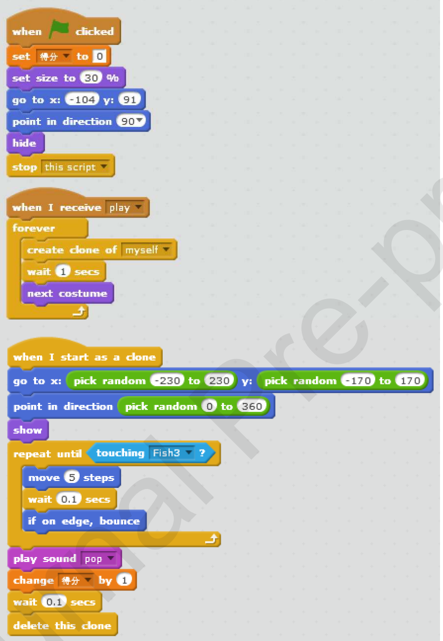
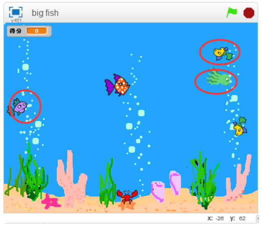
Appendix A: The Course Schedule, Instructional Plan, and Assessments

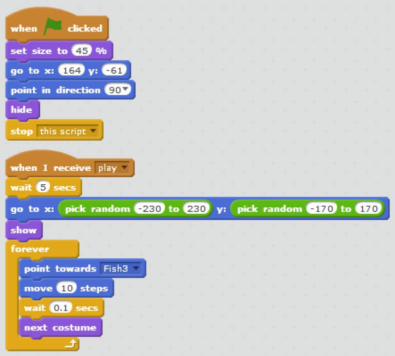
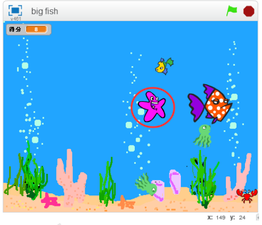
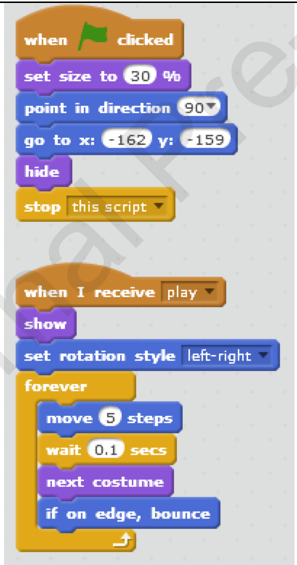
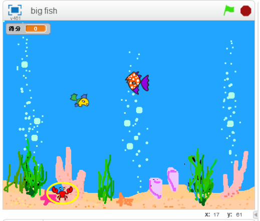
Weeks	Themes /Topics	Programming Tasks	Programming Skills by Students	Assessments and Feedback by Dr. Scratch and the Teacher
1-2	Understanding Scratch	Getting to know Scratch Programming "Happy Fish"	Obtaining hands-on experience with using Scratch; becoming familiar with the functions of each section; preparing for upcoming programming, learning and designing.	Students' work uploaded to Dr. Scratch and assessed by Dr. Scratch (Pre-tests)
3-4	Drawing Pictures	Drawing different shapes Creating spinning windmills	Drawing graphics; learning the concept of "set"; acquiring the concept of calculations — cycles, sequences and so forth; understanding the practice of calculations, e.g., repeat, increment.	Teacher checks students' work and provides feedback
5-6	Inserting Music	Playing Piano Playing beautiful music	Experiencing the principle of using software to control a computer to play music; grasping the concept of computing—loops, conditions, parallelism, etc.; becoming familiar with the practice of computing—repeat, increment.	Teacher checks students' work and provides feedback
7-8	Creating Animati	Creating electronic cards	Experiencing the fun of story creation by writing stories;	Teacher checks students' work

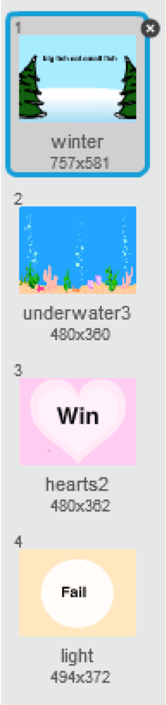
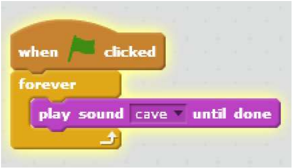
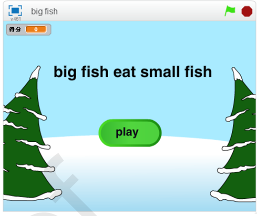
	on	Creating “A Safe Road”	mastering the concept of computing—events, conditions, parallelism, and the process of computing practice—recreation and reuse.	and provides feedback
9-14	Creating Games	Jeopardy “Catching the apples” “Sorting garbage”	Exploring the games by creating works that define goals and rules; mastering the computing concepts: operators, selection conditions, data representation, etc.; becoming familiar with the practice of computing-debugging, testing.	Teacher checks students’ work and provides feedback
15-16	Designing the System	Designing the “Food Chain in the Ocean”	Integrating and applying all the learned computational thinking skills to complete one’s own programming work.	Students’ work uploaded to Dr. Scratch and assessed by Dr. Scratch (Post-tests)

Appendix B: Scratch project entitled “Food Chain in the Ocean” created by the student

Sprites	Programming	Show
 The Play button	 when [flag] clicked switch backdrop to winter go to x: 0 y: -35 point in direction 90 show forever change color effect by 25 wait 0.1 secs when this sprite clicked hide switch backdrop to underwater3 broadcast play stop this script When click the green flag, switch the backdrop to winter picture, set the initial position. When the sprite clicked, hide and switch backdrop to underwater 3, broadcast play, and stop the script.	 When the game starts, music start to play, the play button start changing color until clicked by the player.

 Player	 when 🏴 clicked set size to 40 % go to x: -151 y: -84 point in direction 90▾ hide stop this script ▾ when I receive play ▾ show forever point towards mouse-pointer ▾ go to mouse-pointer ▾ when I receive play ▾ forever if touching Fish2 ▾ ? then change size by 5 wait 0.1 secs if touching Crab ▾ ? or touching Starfish ▾ ? then change size by -5 wait 0.1 secs if size > 150 then say 我赢了！！！ switch backdrop to hearts2 ▾ stop all ▾ if size < 40 then say 我输了！！！ switch backdrop to light ▾ stop all ▾	 The display of the designed game
 1 fish2 132x116 2 octopus-a 141x145 3 fish1 137x108 Three costumes of small fish	 when 🏴 clicked set 得分 ▾ to 0 set size to 30 % go to x: -104 y: 91 point in direction 90▾ hide stop this script ▾ when I receive play ▾ forever create clone of myself ▾ wait 1 secs next costume when I start as a clone go to x: pick random -230 to 230 y: pick random -170 to 170 point in direction pick random 0 to 360 show repeat until touching Fish3 ▾ ? move 5 steps wait 0.1 secs if on edge, bounce play sound pop ▾ change 得分 ▾ by 1 wait 0.1 secs delete this clone When clicking the green flag, set the score to zero, set size to 30%, reset the initial position (104, 91), reset the initial direction, hide and stop the script. When receiving the order "play", clone myself every other second with the next costume. When a clone competes, pick random direction to move, slowly repeat until touching fish 3 until touch the player, play music, change the score by 1 and delete the clone.	 The display of the three costumes of small fish

 The Starfish	 when 🏳 clicked set size to 45 % go to x: 164 y: -61 point in direction 90▾ hide stop this script ▾ when I receive play ▾ wait 5 secs go to x: pick random -230 to 230 y: pick random -170 to 170 show forever point towards Fish3 ▾ move 10 steps wait 0.1 secs next costume When clicking the green flag, set the size of the Starfish to size 45% and set the initial position. Reset the initial direction and hide, stop the script. When receiving the order "play", wait for 5" and show at the random position. Keep moving to the player.	 The interface with the Starfish.
 The Crab	 when 🏳 clicked set size to 30 % point in direction 90▾ go to x: -162 y: -159 hide stop this script ▾ when I receive play ▾ show set rotation style left-right ▾ forever move 5 steps wait 0.1 secs next costume if on edge, bounce When the green flag clicks, set the size of the Crab to 30%, initiate the direction and position, hide the Crab, and stop the script. When receiving the order "play", move the crab and bounce if on the edge.	 The Crab in the game

 1 winter 757x581 2 underwater3 480x360 3 Win hearts2 480x362 4 Fail light 494x372 Pictures of the backdrops	 when clicked forever play sound cave until done When the green flag is clicked, play the sound “cave” until done.	 The Backdrop of the game “Food Chain in the Ocean”

The Effectiveness of Pair Programming on Elementary School Students' Computational Thinking Skills and Self Efficacy

Table 1. A Sample Lesson of an EG Class

Class session	Time	Details
Teacher presentation	10 minutes	The teacher presented the lecture.
Project ideas	5 minutes	Students were paired up to discuss their project ideas and the paired students proposed their ideas together.
Programming in pairs	20 minutes	Students worked in pairs on their project ideas/assignments, including discussing ideas, collecting materials, and helping each other to use Scratch.
Students' project presentations	5 minutes	All students presented their projects, explaining their ideas and design process.

Table 2. A Sample Lesson of a CG Class

Class session	Time	Details
Teacher presentation	10 minutes	The teacher presented the lecture.
Project ideas	5 minutes	Each student came up with their own project idea and proposed their individual idea.
Programming on their own	20 minutes	Each student worked on their own project. When they had questions, they asked the teacher.
Students' project presentations	5 minutes	All students presented their projects, explaining their ideas and programming process.

Table 3. The One-way ANCOVA of Students' CT Skills in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG	84	14.05	3.43	14.06	0.36	10.12^{**}
CG	87	12.45	3.25	12.44	0.36	

$^{**}p < 0.01$

Table 4. The One-way ANCOVA of the Boys' and girls' CT Skills in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG (boys)	50	14.12	3.38	14.12	0.47	6.19*
CG (boys)	50	12.46	3.30	12.46	0.47	
EG (boy-boy pairs)	28	14.57	3.54	14.74	0.63	
EG (boy-girl pairs)	22	13.55	3.14	13.32	0.72	
CG (boys)	50	12.46	3.30	12.46	0.47	
EG (girls)	34	13.94	3.56	13.97	0.58	3.72
CG (girls)	37	12.43	3.24	12.41	0.56	
EG (girl-girl pairs)	12	12.92	3.83	13.10	1.01	
EG (boy-girl pairs)	22	14.50	3.36	14.43	0.73	
CG (girls)	37	12.43	3.24	12.41	0.56	

$^{*}p < 0.05$

Table 5. The One-way ANCOVA of Students' Programming SE in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG	84	62.39	10.89	61.79	1.54	8.37**
CG	87	54.36	13.95	54.93	1.51	

$^{**}p < 0.01$

Table 6. The One-way ANCOVA of the Boys' and girls' Programming SE in the EG and CG

Group	N	Mean	SD	Adjusted mean	Adjusted SE	F
EG (boys)	50	63.20	9.96	63.19	1.99	7.64**
CG (boys)	50	54.64	14.64	54.65	1.99	
EG (boy-boy pairs)	28	62.89	9.35	62.87	2.59	
EG (boy-girl pairs)	22	63.59	10.90	63.58	2.81	
CG (boys)	50	54.64	14.64	54.65	1.99	
EG (girls)	34	61.21	12.18	59.70	2.46	1.35
CG (girls)	37	53.97	13.16	55.35	2.34	
EG (girl-girl pairs)	12	59.42	12.31	56.97	4.04	
EG (boy-girl pairs)	22	62.18	12.29	60.91	2.84	
CG (girls)	37	53.97	13.16	55.35	2.34	

$^{**}p < 0.01$

Table 7. The Correlation between the Students' CT and Programming SE

Group	CT and SE Tests	N	Pearson Correlation coefficient (r)
EG	Pre-tests	84	0.07
CG	Pre-tests	87	0.01
EG	Post-tests	84	0.24*
CG	Post-tests	87	0.19

*$p < 0.05$

Table 8. Main Characteristics of the Four Groups of Students

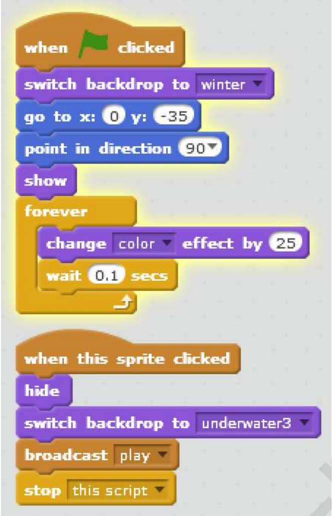
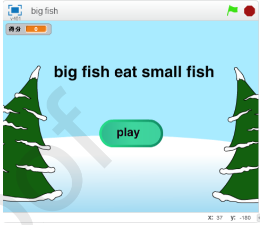
CT	SE	N	Main Characteristics
High	High	4	Confident in solving CT problems; using proper CT language; interested in solving more challenging problems; enjoyed programming collaboratively.
High	Low	4	Confident in ways to solve problems; yet emphasizing continuous improvements in programming; hesitant to share their programming work with others; enjoyed programming with their partners and solving the problems together.
Low	High	4	Lack of CT domain-specific language; yet having no problem looking for solutions to the problems; enjoying collaborative learning experiences.
Low	Low	4	Avoiding discussions of programing processes; discussing the challenges of solving problems; highlighting the enjoyment of working with others.

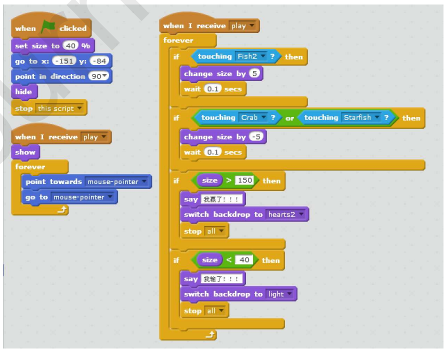
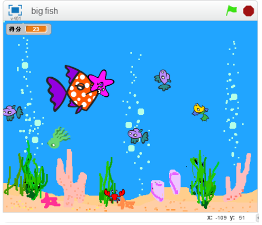
Appendix A: The Course Schedule, Instructional Plan, and Assessments

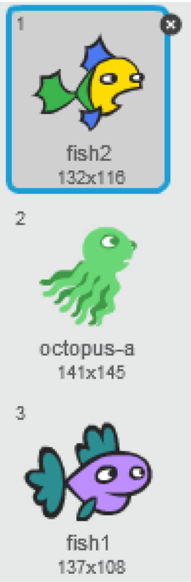
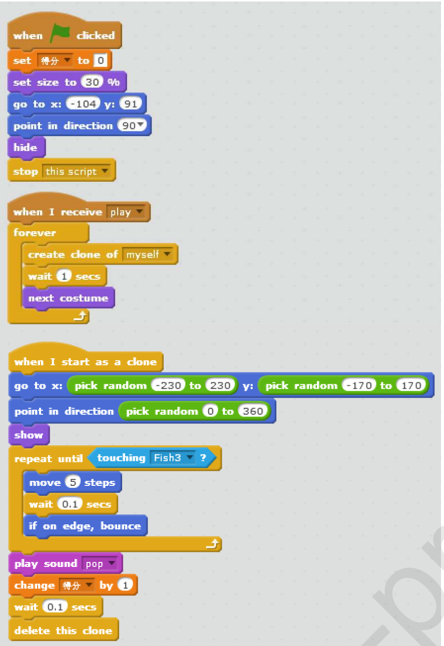
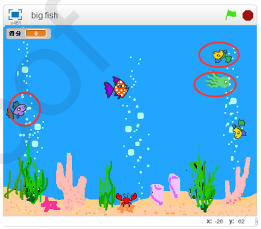
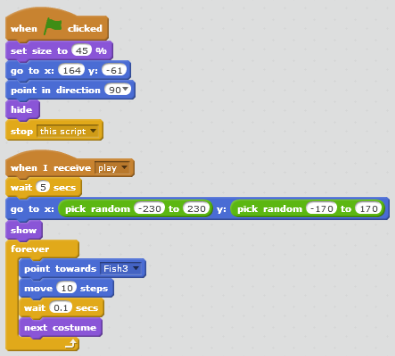
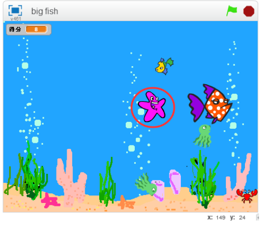
Weeks	Themes /Topics	Programming Tasks	Programming Skills by Students	Assessments and Feedback by Dr. Scratch and the Teacher
1-2	Understanding Scratch	Getting to know Scratch Programming "Happy Fish"	Obtaining hands-on experience with using Scratch; becoming familiar with the functions of each section; preparing for upcoming programming, learning and designing.	Students' work uploaded to Dr. Scratch and assessed by Dr. Scratch (Pre-tests)
3-4	Drawing Pictures	Drawing different shapes Creating spinning windmills	Drawing graphics; learning the concept of "set"; acquiring the concept of calculations — cycles, sequences and so forth; understanding the practice of calculations, e.g., repeat, increment.	Teacher checks students' work and provides feedback

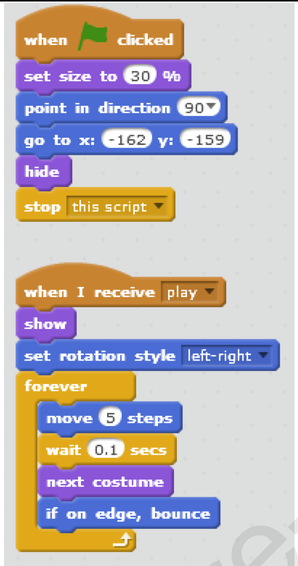
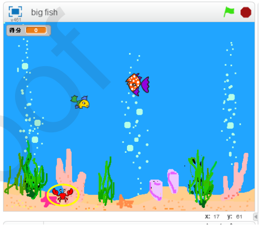
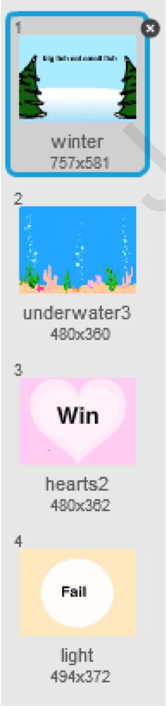
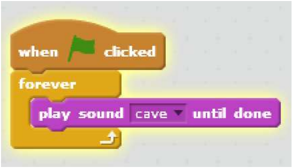
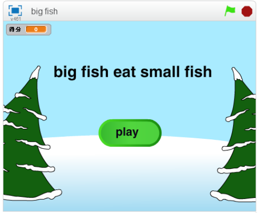
5-6	Inserting Music	Playing Piano Playing beautiful music	Experiencing the principle of using software to control a computer to play music; grasping the concept of computing—loops, conditions, parallelism, etc.; becoming familiar with the practice of computing—repeat, increment.	Teacher checks students' work and provides feedback
7-8	Creating Animation	Creating electronic cards Creating "A Safe Road"	Experiencing the fun of story creation by writing stories; mastering the concept of computing—events, conditions, parallelism, and the process of computing practice—recreation and reuse.	Teacher checks students' work and provides feedback
9-14	Creating Games	Jeopardy "Catching the apples" "Sorting garbage"	Exploring the games by creating works that define goals and rules; mastering the computing concepts: operators, selection conditions, data representation, etc.; becoming familiar with the practice of computing-debugging, testing.	Teacher checks students' work and provides feedback
15-16	Designing the System	Designing the "Food Chain in the Ocean"	Integrating and applying all the learned computational thinking skills to complete one's own programming work.	Students' work uploaded to Dr. Scratch and assessed by Dr. Scratch (Post-tests)

Appendix B: Scratch project entitled “Food Chain in the Ocean” created by the student

Sprites	Programming	Show
 The Play button	 when 🟩 clicked switch backdrop to winter ▾ go to x: 0 y: -35 point in direction 90▾ show forever change color ▾ effect by 25 wait 0.1 secs when this sprite clicked hide switch backdrop to underwater3 ▾ broadcast play ▾ stop this script ▾ When click the green flag, switch the backdrop to winter picture, set the initial position. When the sprite clicked, hide and switch backdrop to underwater 3, broadcast play, and stop the script.	 When the game starts, music start to play, the play button start changing color until clicked by the player.

Player
	
	when 🟩 clicked
	set size to 40 %
	go to x: -151 y: -84
	point in direction 90▾
	hide
	stop this script ▾
	when I receive play ▾
	show
	forever
	  point towards mouse-pointer ▾
	  go to mouse-pointer ▾
	when I receive play ▾
	forever
	  if touching Fish2 ▾ ? then
	    change size by 5
	    wait 0.1 secs
	  if touching Crab ▾ ? or touching Starfish ▾ ? then
	    change size by -5
	    wait 0.1 secs
	  if size > 150 then
	    say 我赢了！！！
	    switch backdrop to hearts2 ▾
	    stop all ▾
	  if size < 40 then
	    say 我输了！！！
	    switch backdrop to light ▾
	    stop all ▾
		
		The display of the designed game

 1 fish2 132x116 2 octopus-a 141x145 3 fish1 137x108 Three costumes of small fish	 when [flag] clicked set 得分 to 0 set size to 30 % go to x: -104 y: 91 point in direction 90 hide stop this script when I receive play forever create clone of myself wait 1 secs next costume when I start as a clone go to x: pick random -230 to 230 y: pick random -170 to 170 point in direction pick random 0 to 360 show repeat until touching Fish3 ? move 5 steps wait 0.1 secs if on edge, bounce play sound pop change 得分 by 1 wait 0.1 secs delete this clone When clicking the green flag, set the score to zero, set size to 30%, reset the initial position (104, 91), reset the initial direction, hide and stop the script. When receiving the order "play", clone myself every other second with the next costume. When a clone competes, pick random direction to move, slowly repeat until touching fish 3 until touch the player, play music, change the score by 1 and delete the clone.	 The display of the three costumes of small fish
 The Starfish	 when [flag] clicked set size to 45 % go to x: 164 y: -61 point in direction 90 hide stop this script when I receive play wait 5 secs go to x: pick random -230 to 230 y: pick random -170 to 170 show forever point towards Fish3 move 10 steps wait 0.1 secs next costume When clicking the green flag, set the size of the Starfish to size 45% and set the initial position. Reset the initial direction and hide, stop the script. When receiving	 The interface with the Starfish.

	the order "play", wait for 5" and show at the random position. Keep moving to the player.	
 The Crab	 when [flag] clicked set size to 30 % point in direction 90 go to x: -162 y: -159 hide stop this script when I receive play show set rotation style left-right forever move 5 steps wait 0.1 secs next costume if on edge, bounce When the green flag clicks, set the size of the Crab to 30%, initiate the direction and position, hide the Crab, and stop the script. When receiving the order "play", move the crab and bounce if on the edge.	 The Crab in the game
 1 winter 757x581 2 underwater3 480x360 3 Win hearts2 480x362 4 Fail light 494x372 Pictures of the	 when [flag] clicked forever play sound cave until done When the green flag is clicked, play the sound "cave" until done.	 The Backdrop of the game "Food Chain in the Ocean"

backdrops		

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self Efficacy

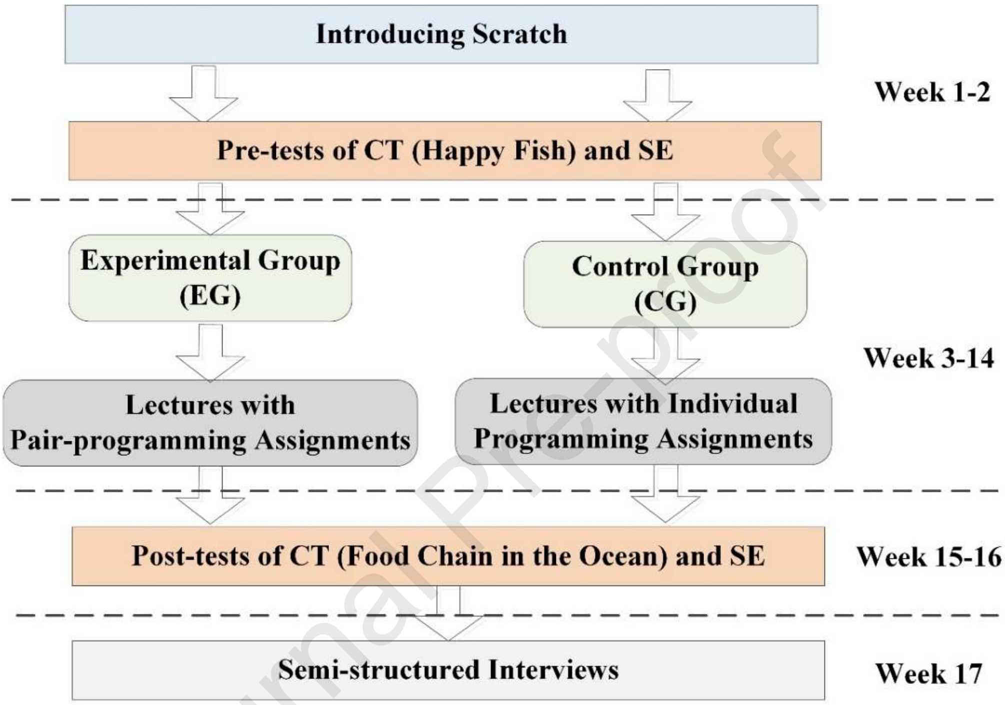
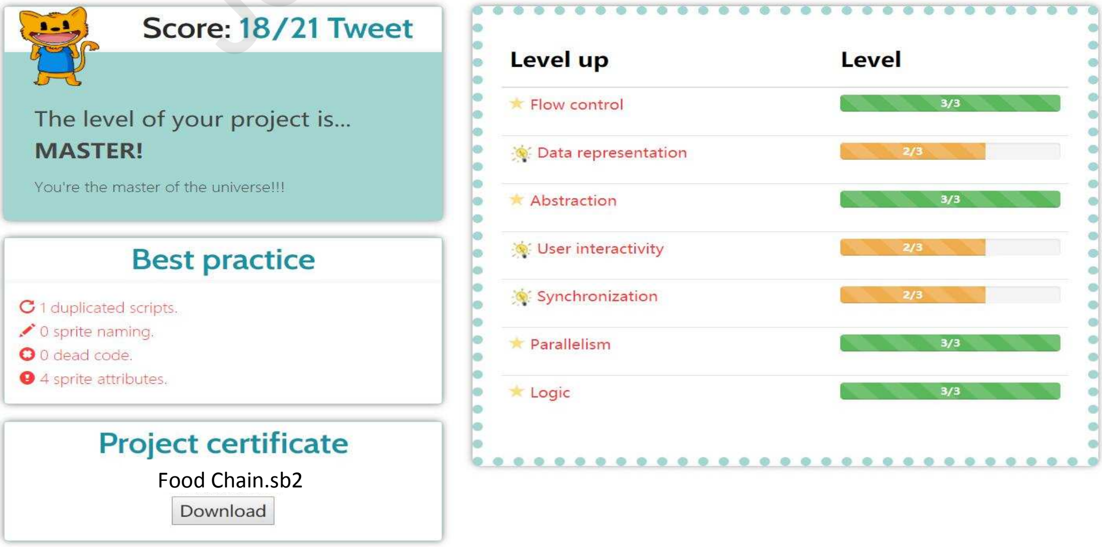

Introducing Scratch
Week 1-2
Pre-tests of CT (Happy Fish) and SE
Experimental Group (EG)
Control Group (CG)
Week 3-14
Lectures with Pair-programming Assignments
Lectures with Individual Programming Assignments
Post-tests of CT (Food Chain in the Ocean) and SE
Week 15-16
Semi-structured Interviews
Week 17

Figure 1. The Experimental Procedure


Score: 18/21 Tweet

The level of your project is...

MASTER!

You're the master of the universe!!!

Best practice

- 1 duplicated scripts.
- 0 sprite naming.
- 0 dead code.
- 4 sprite attributes.

Project certificate

Food Chain.sb2

Download

Level up	Level
Flow control	3/3
Data representation	2/3
Abstraction	3/3
User interactivity	2/3
Synchronization	2/3
Parallelism	3/3
Logic	3/3

Figure 2. An Example of Students' CT Project Assessed by Dr. Scratch

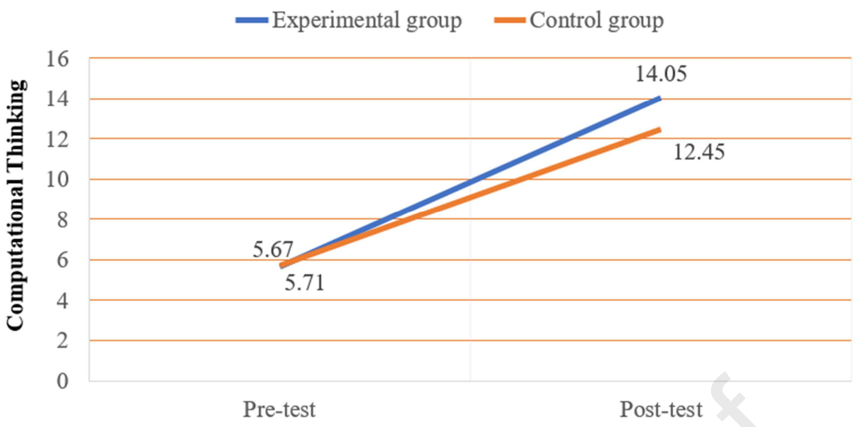


Figure 3. Pre-test and Post-test of CT Scores of the EG and CG students

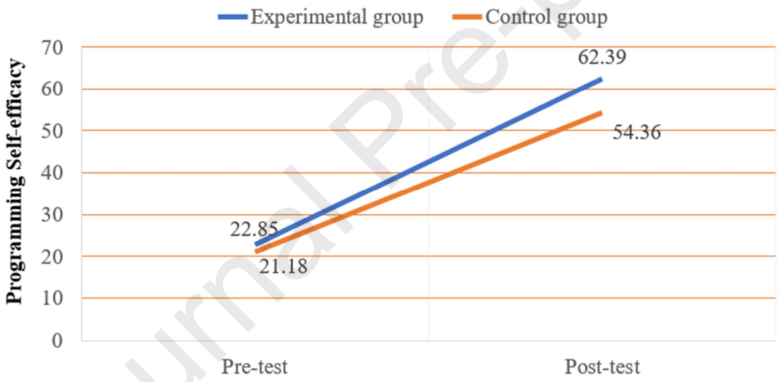


Figure 4. Pre-test and Post-test of SE Scores of the EG and CG Students

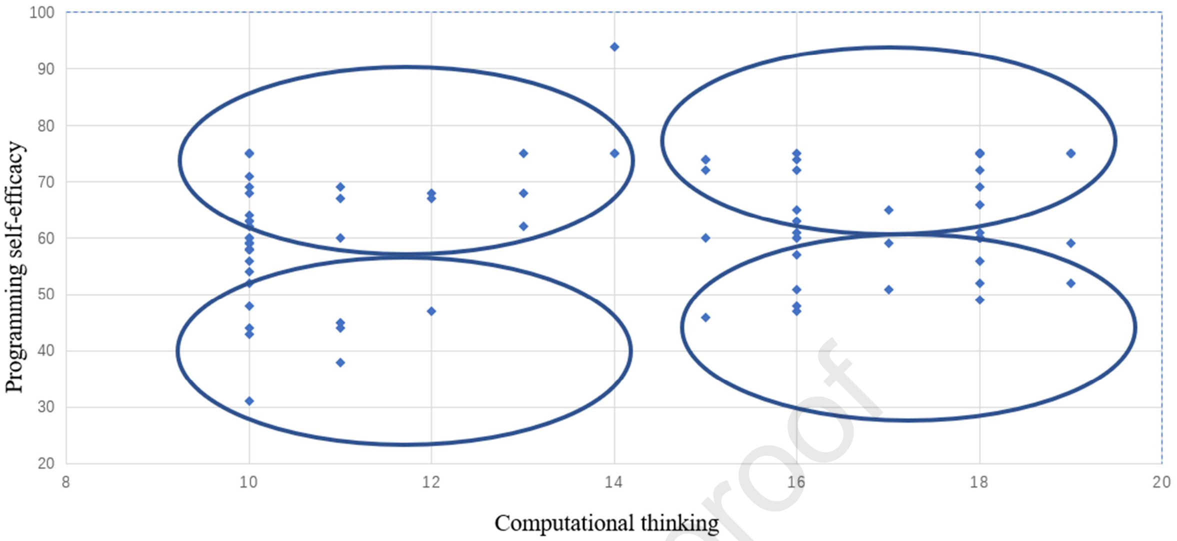


Figure 5. EG Students' Post-CT and Programming SE Scores

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy

Highlights:

- Partial Pair Programming (PPP) significantly improved CT skills of 4th grade students
- PPP significantly improved self-efficacy (SE) of the 4th grade students
- There is low correlation between CT and SE
- PPP significantly improved boys' CT skills in all pairs, girls in boy-girl pairs
- PPP significantly improved boys' programming SE, not girls' SE

The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy

Credit Author Statement:

Xuefeng Wei: Conceptualization, Methodology, Investigation, Writing - Original Draft, Review & Editing, Project administration

Lin Lin: Conceptualization, Methodology, Review & Editing, Supervision

Nanxi Meng: Writing - Original Draft, Review & Editing

Wei Tan: Visualization, Software

Siu-Cheung Kong: Review & Editing

Kinshuk: Review & Editing