JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 22, No. 12, pp. 1977-1985, Dec. 2021

Check for updates

소프트웨어 교육에서 자율적 협동학습이 컴퓨팅 사고력에 미치는 영향

정 혜 연[1] · 서 영 건[2*]
[1]경상대학교 교육대학원 컴퓨터교육전공 학생
[2*]경상대학교 컴퓨터과학과 교수

Effect of Autonomous Cooperative Learning on the Computational Thinking in Software Education

Hye Yeon Jung[1] · Yeong Geon Seo[2*]

[1]Master's Course, Major of Computer Education, Graduate School of Education, Gyeongsang National University, 501 Jinju-daero, Jinju, Gyeongnam, Korea
[2*]Professor, Department of Computer Science, Gyeongsang National University, 501 Jinju-daero, Jinju, Gyeongnam, Korea

[요 약]

교육부에서 컴퓨팅 사고력을 가진 창의·융합 인재를 개발하기 위해 소프트웨어 교육 과정을 운영하고 있으며, 모든 학생들이 미래 사회에서 살아가는데 필요한 컴퓨팅 사고력을 기반으로 문제를 해결할 수 있는 역량을 기르는 것을 목표로 하고 있다. 이에 본 연구는 전통적인 교수-학습법인 강의법과 협동 학습 중 가장 최근에 고안된 자율적 협동 학습 모형을 학습자들에게 적용하였을 때 수업에 대한 흥미도 및 만족도와 컴퓨팅 사고력을 갖추고 문제를 해결할 수 있는 능력에 어떠한 변화를 가져왔는지를 비교해보았다. 두 모형을 실험, 통제집단에 적용한 결과 자율적 협동 학습을 적용한 집단에서 수업에 대한 만족도가 높게 나왔으며 컴퓨팅 사고력을 갖추어 문제 해결 능력이 높아진 것도 확인할 수 있었다. 자율적 협동 학습 모형은 강의법 모형보다 컴퓨팅 사고력을 높이는 데 유의미한 결과를 가져오는 것을 확인할 수 있었다.

[Abstract]

The Ministry of Education operates a software curriculum to develop creative and convergent talents with computational thinking skills, and aims to develop the ability of all students to solve problems based on them necessary to live in the future society. Thus, this study compared how interest and satisfaction in classes, computational thinking skills, and ability to solve problems when applying the autonomous cooperative learning model, compared to the traditional teaching-learning methods to learners. As a result of applying the two models to the experimental and control group, it was confirmed that the satisfaction with the class was high in the group applying the autonomous cooperative learning method, and the computational thinking ability was increased. This is that the Autonomous cooperative learning model had significant results in improving computational thinking ability than the lecture model.

색인어 : 컴퓨팅 사고, 자율적 협동학습, 협동학습, 소프트웨어 교육, 학습 동기

Keyword : Computational thinking, Co-op Co-op, Cooperative learning, Software education, Learning motiveSociety

http://dx.doi.org/10.9728/dcs.2021.22.12.1977

Received 11 October 2021; **Revised** 08 November 2021
Accepted 07 December 2021

***Corresponding Author; Yeong Geon Seo**

Tel: +82-55-772-1392
E-mail: young@gnu.ac.kr

Ⅰ. Introduction

Today, with the development of information and communication technology, we have arrived in the era of the 4th industrial revolution. As an IT technology powerhouse, Korea is at the center of the 4th industrial revolution, achieving rapid development in all fields, including industry, technology, finance, medical care, and education, and is standing at the center of it. The lives of modern people affected by rapid development are getting closer to popular smart devices every day, and the future generation feels the need to prepare digital devices necessary for human life and convergence technology with various fields. Therefore, the rapidly changing information society of the future requires talents with new competencies with different education than before the 4th industrial revolution. Until the 20th century, the 3rd industrial revolution focused on computer dissemination, and it was important to think about hardware and simple computer usage ability. However, unlike before, we, who are living in the 4th industrial revolution and the 21st century, need to think about what kind of talent we want in the future. According to the social flow, interest in software is growing and companies are developing IT-related fields from existing business fields. In line with this trend, the countries around the world recognize the importance of early software education and computer literacy and have established themselves as a mandatory course.

According to [1], computing subjects were created for all educational stages of 5 to 16 years old to learn programming, and in the US, K-12, computer science standards from the(CSTA: Computer Science Teachers Association) provides educational directions for computer education and software education. Of course, in Korea, software education is made compulsory in elementary and middle school education courses to discover talented people with computational thinking skills. The Ministry of Education[2] revised the curriculum in 2015 and strengthened it as an information curriculum that emphasizes the role of software education in the entire school education process to foster creative convergence talents, which is the government's main educational reform task. And it has a goal of nurturing talented people who can prepare for the future society by developing computational thinking skills centered on software, design systems on their own, and actively solve problems. The biggest feature of the 2015 revised curriculum is that it introduces software education while maintaining the subject and grade groups of the 2009 revised curriculum. However, there are still problems with the educational environment and methods for enhancing information science thinking and computational thinking. In general, software education uses computers or information and communication devices to learn and develop problem-solving skills, but it requires efforts from other educators.

Therefore, this study intends to explore the future direction of information education by experimenting with two teaching and learning methods as a way to improve computational thinking at the beginning of software education.

Ⅱ. Related Works

2-1 Computational thinking

In our daily life, we have to analyze a lot of data to solve problems in complex fields, and when we want to organize the problems with a computer and find the principles, we use computational thinking. This ability refers to the thinking ability that students who are accustomed to simple learning can efficiently solve various problems that can occur in daily life with the concepts and principles of computing. Basically, it consists of decomposition, automation, pattern recognition, algorithm, and abstraction.

Decomposition means breaking up a problem into smaller, manageable units, including data and processes. Automation refers to making commands to be given to a computer in a computer language or code that the computer can understand to solve a problem, and it refers to programming to solve problems and the entire process of executing them. Pattern recognition is to predict and solve problems by finding repeating properties or rules in a certain problem or phenomenon. An algorithm is a step-by-step procedure or process that sequentially expresses a series of steps to solve a problem. Abstraction refers to the process of establishing general principles for creating patterns and expressing data through modeling or simulation.

According to [3], in the Korea Education and Research Information Service(KERIS), computational thinking skills have confidence in dealing with complex problems, perseverance in dealing with difficult problems, tolerance for ambiguity, the ability to deal with open-ended problems with unanswered questions on their own, and other factors to achieve a common goal or solution. It is said that it supports and develops talents with the ability to share opinions, communicate and work with people. In the 2015 revised information curriculum, computational thinking ability is the ability to understand real life and various academic problems by using the basic concepts and principles of computer science and computing systems, and to implement and apply creative solutions[2].

2-2 Software education

The major improvement by school level of the 2015 revised curriculum is to strengthen software education compared to the 2009 revised one to understand various problems in real life by experiencing algorithms and programming based on a sound information ethics awareness[2]. If the previous software education operation guidelines focused on the method of writing documents or materials with application programs, the software education of the revised curriculum is based on the use of play, experience, and educational programming languages that can enhance computational thinking and problem solving skills. It is an education that can develop convergence skills such as thinking, logic, and problem solving skills with problems that can be solved in real life such as sequential structure, repetition structure, and choice structure.

The software education presented by the Ministry of Education is designed to enable learners to recognize the meaning and importance of computational thinking, an essential element of a digital society, and to check its value through performance-oriented education rather than knowledge-oriented education. First, the standard of achievement that can be hoped for in software education in elementary school can be expressed as a change in the overall lifestyle of society, which has changed from ‘life and software’ to software, etiquette to keep in cyberspace, internet addiction, and protection of personal information and copyright. Next, in the 'algorithms and programming' area, students understand and simplify the problems presented and explain how to solve the problems in order to explore solutions, and solve problems using sequential, selection, and repetition structures for the concept of algorithms to indicate the procedure. Also, they can know the basic elements of a programming language, and come up with ideas and write their own programs.

In the middle school, the achievement standards desired for learners are the types and characteristics of software in the area of 'life and software', and the concepts, types, damage cases, and prevention methods of personal information applied and used in real life. In addition, students can understand computer components, functions, types and characteristics of operating systems, and networks. In the next 'algorithm and programming' area, the concepts of data and information and types of information are distinguished and the concepts of linear and non-linear structures are understood to structure information, and in the form of lists, hierarchies, tables, diagrams, etc. In addition, it is possible to analyze, structure, and abstract to solve a problem, to explore various solutions with a strategy, and to design an algorithm for solving a problem by understanding the conditions that an algorithm must have. After that, they can learn about various programming languages and about the development environment and programming basic structure. In the last ‘computing and problem solving’ area, real life problems and problems in various areas can be solved with algorithms and programming.

In the last high school, the expected achievement standards of learners are explained by learning about computing technology, convergence, the future of software, information ethics and intellectual property protection, response technology, operation principle, and information processing in the 'life and software' area. In the area of 'algorithms and programming', they learn about information representation methods, management, problem structuring, abstraction, modeling and simulation, designing, analyzing, evaluating, programming procedural and object-oriented languages of complex structures, and distinguishing basic structures. In the final ‘computing and problem solving’ area, they can design, build, and evaluate projects as a team, understanding programming used in multiple disciplines.

2-3 Understanding cooperative learning

When several people come together to achieve a single goal, it is called ‘cooperation’. Humans have been living organically by joining forces with each other to form a society to survive. We work together by adding a small amount of power to achieve one worthy goal, and we live together for the sake of everyone. According to [4], cooperative learning is a teaching method in which class students work together to achieve a single learning goal. Cooperative learning has several elements that distinguish it from other forms of learning.

- Having a common goal
- All learners being on an equal footing
- Emphasis on personal accountability
- Producing the results that are beneficial to all.
- Shared responsibility for results

According to [5], the essence of cooperative learning is to allocate a group goal so that a common achievement goal can be reached, to raise the average grade of the group, and to reward the group as a whole by reviewing the quantity and quality of grades according to a set standard. This is referred to as a series of teaching-learning procedures. In cooperative learning, [5] shows an attitude of 'the whole is for the individual (all-for-one) and the individual is for the whole (one-for-all)', and encourages and supports each other for the successful learning of group members. It was said that learning difficulties could be improved by giving.

Cooperative learning and collaborative learning are often confused in terms. Cooperative learning is to increase positive

interdependence among students, clarify personal responsibility, and ensure that all students participate in class at the same time so that no students are left out. Collaborative learning is effective when students have some basic learning ability, but it is not suitable for students who lack the will to learn. It is the role of the teacher to determine the appropriate class by distinguishing between cooperative learning and collaborative learning in consideration of level, will, and situation of the students.

There are several instructional models of cooperative learning as follows : STAD(Student Teams Achievement Divisions), TGT(Teams GamesTournament), Jigsaw Ⅱ model, TAI(Team Assisted Individualization) model, LT(Learning Together), Jigsaw Ⅰmodel, GI(Group Investment) model and autonomous cooperative learning (Co-op co-op) model. Co-op co-op model is a relatively recent 'cooperative learning model for cooperation' based on the cooperative learning model. It is very similar to the group inquiry model, and it is a cooperative learning type in which the teacher presents the learning topic and goal to the whole class and performs cooperative learning in small groups by dividing sub-themes to solve the goal. When forming a group, students who are interested in the same sub-topic gather and divide into mini-topics within each group to learn the part they are assigned to. When learning is complete on a mini-topic, small groups come back together to share content and achieve the learning goals given to the entire class[6]. Co-op co-op model was developed to supplement the group inquiry model. Table 1 is summarized by [7] to compare the model and the group inquiry model.

Kagan explained that the true role of education is to provide conditions for the development of students' natural curiosity, intelligence, and expressiveness[8], and Jung explained that students themselves share the same topics with interest[6]. Seo said that Co-op co-op model can enhance individual accountability by allowing students to form groups on their own and participate autonomously in preparation, learning, and evaluation reports, and can also increase their ability to communicate to reconcile their differences[9]. A learning environment should be provided for discussion, and research on a topic with interested colleagues. To successfully operate Co-op co-op model, the 10-step procedures in table 2 must be included[10].

2-4 Problems and Comparisons with Existing Studies

[11] emphasizes individual responsibility in situations of various levels of learners, and utilizes the Achievement Task Sharing Learning Model (STAD), in which learners of various abilities can help each other and receive help to achieve a common goal, and analyzed how much it affects the improvement of problem solving ability. In the study, traditional teaching method was applied to the control group and unplugged learning of the achievement task sharing learning model was applied to the experimental group. There was a statistically significant improvement in computational thinking, but there was no significant difference in cooperative problem solving ability.

[12] conducted a class based on STAD model, and assumed that the group that performed cooperative learning took less time to solve the problem than the group that performed individual learning, and the problem solving accuracy was also excellent. In conclusion, it was said that the problem solving ability of the group to which cooperative learning was applied was more effective, and that students shared their thoughts through cooperative learning.

In [13], they conducted that online cooperative learning had a more positive effect than the traditional classroom method with programming education using online cooperative learning in terms of academic achievement change. In other words, it was said that online cooperative learning had a positive effect on academic achievement and creative thinking ability in programming education than conventional programming education.

Most of the existing studies that have conducted software education so far have focused on changes in computational thinking ability. There were very few studies on specific teaching methods in software education, and among them, fewer studies related to cooperative learning. As a software education, there have been many studies that use various teaching aids such as robots and unplugged to teach and focus only on changes in computational thinking skills. However, as the information subject became a compulsory subject for middle school, the change in computational thinking ability according to the teaching materials is also important, but like other subjects that were previously compulsory subjects, we are trying to study how cooperative learning applied in software education can have an effect on students according to teaching and learning methods.

표 1. 집단 탐구 모형과 Co-op co-op 모형의 비교
Table 1. Comparison of group inquiry model and Co-op co-op model

Supplementary items	Group inquiry model	Co-op co-op model
Segmentation of learning process	6-step instructional process	10-step instructional process
Active reflection of learner's opinion	Formation of sub-themes through categorization of questions between teachers and students	Formation of sub-themes through active student-centered discussion and discussion
Diverse opportunities to choose	Choosing sub-topic	Choosing a sub-topic and choosing a mini-topic within a sub-topic
Strengthening individual accountability	Individual collection and learning optional	Individual collection and learning essential

표 2. Co-op co-op 모형의 단계별 활동 내용
Table 2. Step-by-step activity contents of Co-op co-op model

Step	Contents of class
1 step	Introduction to learning topics
2 step	Student-centered class discussion
3 step	Selecting sub-topic
4 step	Organizing groups by sub-topic and refining sub-themes
5 step	Mini-topic selection and division of labor
6 step	Individual study and group presentation preparation
7 step	Presentation of the mini-topic within the group
8 step	Preparing for class presentation by group
9 step	Class presentation and discussion by group
10 step	Evaluation and reflection

Ⅲ. Research Approach and Procedure

3-1 Research problem and its procedure

Teaching and learning methods that teachers can use in software education include lecture method, demonstration and practice, cooperative learning, problem-centered learning, discussion method, role play, discovery learning, and apprenticeship. Co-op co-op model, which is the most recently devised cooperative learning model for cooperation, was selected. The research problems of this study are as follows.

First, does the software education teaching method using Co-op co-op have a significant effect on increasing students' learning interest and satisfaction?

Second, does the software education teaching method using Co-op co-op have a significant effect on the expansion of computational thinking compared to the traditional teaching and learning method?

The procedure of this research proceeds in the order of research topic selection, research experiment planning, research subject analysis, research experiment, and research result analysis.

First, the subject of this study was to investigate computational thinking among the various competencies of learners that can be cultivated through software education and to investigate various teaching and learning methods. Among the investigated models, the study was selected as the topic because there were not many studies that simultaneously applied and compared the lecture method, which was mostly conducted in the existing software education field, and Co-op co-op method, which was rarely used.

Second, for the research experiment plan, a lesson plan was prepared according to the lecture method and Co-op co-op model procedure, and the measurement test tool that can evaluate computational thinking ability was examined and the learner's changes were observed.

Third, the subject of this study was the learners of two classes of free grade system in the first year of H middle school located in Geoje-si, Gyeongsangnam-do. Most of the research target group experienced only basic learning such as the experience of problem solving in algorithm and programming area of elementary school, the experience of algorithms, and the experience of programming among the curriculum revised in 2015. It consists of students who are interested in the class or who have chosen to learn.

Fourth, based on the data analyzed in the research experiment, a learning process plan is drawn up, and the second class of each group is conducted based on the learning process plan. And the results of pre test and post test of learners' satisfaction with class and computational thinking ability are investigated.

Finally, after class satisfaction and the effects of differences in teaching and learning methods were compared, the research results were analyzed. There are four limitations of this study. First, this study is conducted for learners in two classes of free grade system. Due to the nature of research targeting learners in a specific region, there are limitations in generalizing the research results. Second, there is a gap in their interest in software classes because the learners who were the subject of this study select the program and the free grade system class voluntarily or involuntarily. Third, this study is a period of global disasters due to the spread of Corona-19. The guidelines of the Ministry of Education and the steps of government and local social distancing have changed, so free-grade students were not able to go to school continuously. Because the changes in students' competency could not be observed consistently, there is a limit to the application of the presented research results during the regular curriculum. Fourth, as the local infection of Corona-19 spread when this study was conducted, students self-quarantine increased, so the number of attendance at each class was changed, and the number of people changed each time the study was conducted.

표 3. 연구 대상 분포
Table 3. Distribution of research subjects

Division	Experimental group	Control group
Attendance	19	17
Pre test (mean)	48.42	54.71
Pre test (sd)	23.396	21.248
t-value	0.840	
p-value	0.407	

Table 3 shows the distribution of research subjects, and is the result of an independent sample t-test to check the homogeneity in advance. To find out what the statistical significance of the result is, the significance probability p-value was calculated. In the pre test, the two groups had a t-value of 0.840 and a p-value of 0.407 under the assumption of equal variance, which was greater than the significance level of 0.05.

3-2 Inspection tool

To investigate the satisfaction(qualitative) with software education, a post-interview was conducted. Interviews were conducted for the experimental group and the control group, respectively, and the questionnaire developed by KERIS was modified to ask questions about interest in programming and questions about research teaching methods.

According to Kim[14], computational thinking is defined as finding and solving problems on the premise of solving problems using a computing system. Therefore, in order to evaluate the effectiveness(quantitative) of the teaching and learning method presented in this study, the problem solving ability test tool and the activity sheet created based on the problem solving stage instructional design were evaluated to measure the problem solving stage ability. To measure the change in computational thinking ability before and after class, it was modified and supplemented according to this study using the information science thinking-based problem solving ability evaluation criteria of Kim[15].

표 4. 문제 해결 단계별 창의적 사고 요소 구분
Table 4. Classification of creative thinking elements in each stage of problem solving

Division	Questions	Creative thinking elements
Problem analysis and expression	1, 6, 7	Sophistication, sensitivity, reconstitution
Problem solving method	4, 8, 9	Fluency, flexibility, originality
Problem solving design	2, 3, 5, 10	Sophistication, reconstitution

According to table 4, if the questions on the test paper are structured and analyzed according to the problem solving stage, first, questions 1, 6, and 7 correspond to the 'problem analysis and expression' stage, and questions 4, 8, and 9 correspond to the 'problem solving method'. The 2, 3, 5, and 10 questions correspond to the 'problem solving design' stage[16]. As the evaluation criteria for each stage of problem solving used in this study, the activity sheet was evaluated using three evaluation criteria for each stage.

3-3 Applying teaching and learning model

A traditional teaching and learning method, lecture method, is the most common method of unilaterally imparting knowledge to learners. It is the oldest and most widely used method of teaching, and demonstrating the knowledge the teacher has. In general, it follows the order of attention, motivation, presentation of learning goals, explanation of concepts and activities, and organizing. In addition, classes can be conducted according to Gagné's 9 class situations and Osbell's meaningful learning theory presented in instructional design theory.

The learning process proposed in this study made it possible to proceed with the class by applying the order of the general lecture method. The learning process plan of Co-op co-op applied the 10 step procedure described above to form a sub-topic through active discussion centered on the student, and according to the sub-theme, the students could select a desired topic and form a small group. Table 5 shows 2nd learning process plan for the experimental group.

표 6. 집단별 프로그래밍에 대한 관심도
Table 6. Interest about programming by group

No	Question	Control group	Experimental group
1	How difficult is the coding class?	2.95	3.13
2	How interested are you in the coding class?	3.55	4.00
3	I think I learned what coding is through this class.	3.50	3.88
4	I think the coding class helped my creativity.	3.50	4.13
5	I think the coding class helped me to improve my problem solving skills.	3.50	3.94
6	I think that the coding class helped improve the convergence thinking ability.	3.41	4.00
Average		3.40	3.85

Ⅳ. Research Results and Analysis

4-1 Class satisfaction

Table 6 is the interview result related to the degree of interest in programming by modifying the questionnaire developed by KERIS. This questionnaire was made on a 5-point scale and was surveyed once after every class in which each teaching and learning method was applied. In the interview conducted to investigate the satisfaction of the class, the average of the control group was 3.40 and the average of the experimental group was 3.85. It was found that the interest in software education taught through Co-op co-op was higher than that of traditional lecture methods, and satisfaction was high.

표 5. 실험 집단 2차 학습과정안
Table 5. 2nd learning process plan for experimental group

Learning Objectives	-Can understand the repetition structure. -Can explain examples of repeating structures.		
Learning material	Teacher		
	Presentation material, Activity sheet		
Step	Learning process	Teaching and learning activity	Time
Introduction	- Greetings and attention - Small group activity notice - Motivation - Introduction to learning topics - Student-Centered Class Discussion	- After greeting, check attendance and call attention. - Explain the sequence structure and daily routine learned in the previous lesson again. . Preparation for class, school life, schedule after school, preparation for bed, etc. - Foretell small group activities. - Describe the academic calendar for one day, one week, one month, and one year. - As a presentation material, present the learning objectives for today's lesson. . Concept of repetition structure . Example of repeating structure	5 min
Deployment	- Preparing small group activities - Selecting sub-topic - Organizing groups by sub-topic and refine sub-themes - Mini-topic selection, division of labor, and individual study - Preparing group and class presentations	- Distribute individual activity sheets. - Write the guessed concept of 'repeat' on the activity sheet. - Ask and answer questions about the various types of repetition experienced in life. - Write the guessed concept of 'repetition structure' on the activity sheet. - Ask and answer examples of machine executing instructions in a repeating structure - Create a repeating structure with a scratch program and select the project topic you want to express. - For each topic, form a group to form a group of 3 to 4 students. - After distributing the small group activity sheet, come up with an idea so that all members can participate and record it on the activity sheet.	20 min
Evaluation and conclusion	- Class presentation and discussion by group - Organize - Preliminary notice - Greetings	- Take turns giving small group presentations, and if there are any errors or corrections, criticize and correct them together. - Ask questions about parts that are difficult to understand. - Foretell the content to be learned next time, - Greet the students and organize the class.	20 min

In table 7, the control group had an average score of 3.16, and the experimental group in which the class was taught in Co-op co-op scored an average of 3.34 points, indicating that learners were more interested in the experimental group. The students in the control group, who taught the class using the lecture method, said, “It was good to understand because you explained how to program in detail.”, “I found out that sequential and repetitive structures are used a lot in life.” with answers such as, “It was easy to understand when I took the class, but since I was going to program myself, I had no idea where to start.” The experimental group, who conducted the class with Co-op co-op, said, “It was easy to understand because I was able to program with a friend who had similar thoughts.” with answers such as, “It is the same topic, but it was difficult to discuss with the group members when organizing the presentation.”

4-2 Effect of Co-op co-op on computational thinking

In the tests conducted before and after each second class with the lecture method and Co-op co-op, the pre test results of the two groups were 54.71 and 63.50, respectively, and the average post test results were 48.42 and 59.44, respectively. So, there was a significant difference in problem solving ability before and after class. This test sheet is made up of 100 points.

The results of the pre and post test mean scores of two groups for each problem solving stage are as follows. Both the groups received the highest score in the 'problem analysis and expression' stage, where necessary data could be found by subdividing and interpreting the problem and solving problems in various ways of expression. It received the lowest score in the 'problem solution design' stage, where the optimal solution was found and expressed in logical steps. It seems that students who are not familiar with programming and who are getting software education for the first time have the most difficulty finding solutions to solve their own optimal problems.

An independent sample t-test was performed to verify whether there was a significant difference in computational thinking according to the teaching and learning method. The t-value of the two groups was 0.550 and the significance probability p-value was 0.586, which was larger than the significance level ($p<.05$), showing a significant difference. As a result of conducting an independent sample t-test based on the mean of pre and post test of the control group, the pre test mean was 54.71 and the post test mean was 63.50, which increased by 8.79 points. T-value was -1.148, and the significance probability p-value was 0.259, which was larger than the significance level of 0.05, which was analyzed to be significant in improving computational thinking.

표 7. 집단별 수업 만족도
Table 7. Class satisfaction by group

No	Question	Control group	Experimental group
1	I can understand and use the basic statements of programming languages.	3.14	3.25
2	I can understand and use sequential structures in programming languages.	3.18	3.19
3	I can understand and use the repetition structure of programming languages.	3.18	3.25
4	By understanding the repetition structure of programming languages, I can easily handle duplicate commands or implement them in a new way.	3.14	2.94
5	I can implement simple functions in a programming language.	2.95	3.44
6	I can solve simple problems with programming languages.	3.05	3.44
7	I like to solve complex computer programming tasks faster than others.	3.27	3.19
8	I can see new problems and solve them using computer programming languages.	3.09	3.06
9	I have experience in effectively processing complex data with computer programming languages.	2.82	2.56
10	I have experience creating works using computer programming languages.	2.95	3.88
11	I've seen other people solve big problems with computer programming languages.	2.82	3.13
12	I've seen my friends make art in computer programming languages.	3.00	3.75
13	I have heard that learning a computer programming language will be very good for learning and living in the future.	3.59	3.75
14	I have a feeling it would be great to learn a computer programming language.	3.68	3.75
15	I can vaguely tell other students that learning computer programming languages is good.	3.32	3.19
16	I want to know what you don't learn when you learn a computer programming language.	3.28	3.31
17	When I learn a computer programming language, I learn without passing time.	3.09	3.38
18	When I learn a computer programming language, I enjoy using the different features.	3.32	3.69
19	I am motivated to extend the capabilities of the computer programming language I have learned to a higher level.	3.14	3.56
20	I also want to learn and use the functions used by other students when learning the computer programming language I have learned.	3.23	3.69
21	I want to make software better than other students.	3.14	3.50
22	I spend more time thinking about computer programs than other students.	2.86	2.63
23	I do one thing until I finish it.	3.50	3.31
Average		3.16	3.34

As a result of the experimental group, the pre test mean was 48.42 and the post test mean was 59.44, which increased by 11.02 points. T-value was -1.532 and the significance probability p-value was 0.134, which was larger than the significance level 0.05 value, which was analyzed to be significant in improving computational thinking ability.

표 8. 집단별 컴퓨팅 사고력 검사 결과
Table 8. Test results of computational thinking by group

	Division	N	Mean	SD	t-value	p-value
Pre test	Control group	17	54.71	21.248	0.840	0.407
	Experimental group	19	48.42	23.396		
Post test	Control group	20	63.50	24.767	0.550	0.586
	Experimental group	18	59.44	20.138		
Test result between pre and post test of control group					-1.148	0.259
Test result between pre and post test of experimental group					-1.532	0.134

표 9. 문제 해결 단계별 실험 결과
Table 9. Test results by steps of problem solving

Division	Control group		Mean	SD	Experimental group		Mean	SD
	Pre	Post			Pre	Post		
Problem analysis and presentation	60	74	67.0	7.0	52	75	63.5	11.5
Exploration of problem solving	68	58	63.0	5.0	61	51	56.0	5.0
Design of problem solving	40	58	49.0	9.0	36	52	44.0	8.0
Total Average	56.00	63.33			49.67	59.33		

Ⅴ. Conclusion

Although research is being conducted on various methods to increase computational thinking, active research is being conducted on how to effectively apply software education to increase computational thinking through coding and programming in school settings. This study compared the problem solving ability when using the traditional teaching and learning method, which is generally used most in the school field, and the most recent cooperative learning model, Co-op co-op method. This model cannot be said to be the optimal way to increase computational thinking, and more research cases applying various cooperative learning models will be needed in the future. In addition, we think that great efforts are needed to enhance computational thinking as the most suitable teaching and learning method in future software education by applying various models with a broad perspective on teaching and learning methods as well as cooperative learning models to increase computational thinking. In this study, after class was planned

according to Co-op co-op model, the class was actually conducted at the school site, and the meaning of problem solving ability according to Co-op co-op and computational thinking ability was investigated. However, since it was not possible to test under the same conditions every time due to various limitations, a follow-up study should be conducted.

References

[1] Y. Kim, Development and Application of Elementary School Software Education Programs with Flip Teaching Method, *Seoul National University of Education, Graduate School of Education*, 2016.

[2] Software Training Operating Guidelines, *Ministry of Education*, 2015.

[3] S. Kim, Effect of Software Education Based on Physical Computing on Elementary School Students' Computational Thinking, *Gyeongin National University of Education, Graduate School of Education*, 2019.

[4] J. Choi, The Effects of Environmental Art Classes Applying Co-op Co-op Model on Second Graders of Elementary School Student's Environmental Literacy, *Ehwa Women University,Graduate School of Education*, Feb. 2020.

[5] Y. Kim, Effect of STAD Cooperative Learning on the Academic Performance and Learning Attitude of Department of Beauty, *Graduate School of Education Konkuk University*, Aug. 2012.

[6] M. Jung, Understanding and Practice of Cooperative Learning, *Education Science Company*, 2006.

[7] J. Han, A Study A Study on Teaching Method for Design Area in the 1st Grade of High School - Focusing on a Cooperative Learning Model, *Ehwa Women University, Graduate School of Education*, Feb. 2007.

[8] S. Kagan, Cooperative Learning. *San Clemente: Resources for Teachers*, 1992.

[9] H. Seo, A Study on Teaching Mathematics Applying the Cooperative Learning Model Co-op Co-op − Focusing on teaching proofs of Properties of Parallelogram in Second Grade Middle School Curriculum, *Hanyang University, Graduate School of Education*, Aug. 2010.

[10] M. Jung, Understanding and Practice of Cooperative Learning, *Education Science Company*, 2002.

[11] M. Seo, The Effects of Unplugged Learning Based on Collaborative Learning on the Improvement of Collaborative Problem Solving Ability and Computational Thinking Ability in Elementary School Students, *Korea Teacher's University, Graduate School of Education*, Feb. 2020.

[12] I. Gwak, A Study on the Effects of Cooperative Learning on Algorithm Concept Learning, *Korea University, Graduate School of Education*, Aug.. 2020.

[13] D. Kim, Effect of Programming Education through Online Cooperative Learning on Academic Achievement and Creative Thinking Ability, *Gongju University, Graduate School of Education*, Feb. 2021.

[14] J. Kim, "*Problem Solving Ability Based on Computational Thinking Ability*", Journal of Information Processing Society, Vol. 24, No. 2, pp. 13-21, Mar. 2017.

[15] J. Kim, Secondary Education Program for Problem-solving Ability based on Computational Thinking, *Korea University, Graduate School of Education,* Feb, 2009.

[16] M. Heo, A Study on Instructional Design Methods to Improve Problem Solving Ability in Information Education, *Korea University, Graduate School of Education,* Aug, 2009.

정혜연(Hye Yeon Jung)

2019년～현 재: 경상대학교 교육대학원 컴퓨터교육전공 석사과정

※관심분야 : 컴퓨터 교육, 컴퓨팅 사고, 인공 지능

서영건(Yeong Geon Seo)

1987년 : 경상대학교 전산과(이학사)
1997년 : 숭실대학교 전산과(공학박사)
1989년～1992년 : 삼보컴퓨터
1997년～현 재 : 경상대학교 컴퓨터과학과 교수
2011년～현 재 : 경상대학교 공학연구원
2018년～2020년 : 경상대학교 전산정보원장

※관심분야 : 의료 영상 처리, 머신 러닝, SLAM, 컴퓨터 네트워크, 컴퓨터 교육