Universal Journal of Educational Research 7(5A): 42-53, 2019 http://www.hrpub.org
DOI: 10.13189/ujer.2019.071507

Development and Application of Programming Education Model Based on Visual Thinking Strategy for Pre-service Teachers

Young-Hoon Sung[1], Young-Sik Jeong[2,*]

[1]Department of Computer Education, Chinju National University of Education, Korea
[2]Department of Computer Education, Jeonju National University of Education, Korea

Abstract In the software-oriented society, the importance of computational thinking for problem solving is increasing day by day. In particular, programming is a good way to improve this kind of computational thinking, and beginners are having difficulties in creating and programming procedures for problem solving. In this study, we developed a Thinking-Revise-Encoding-Evaluating (TREE) model based on a visual thinking strategy for pre-service teachers and applied a 15-week curriculum. The results of the study showed that pre-service teachers' SW education perception, block programming ability, computational thinking, and satisfaction were significant.

Keywords SW Education Model, Visual Thinking, Computational Thinking, Education System, Programming Curriculum

1. Introduction

The rapid development of technologies such as Big Data, Artificial Intelligence, and the Internet of Things are core technologies that lead the fourth industrial revolution. With these changes, our lives need the ability to solve new problems every day. These technologies are largely composed of software that can be controlled with hardware that is miniaturized and processed at a high speed, and in particular, the role of software is increasing. Therefore, in the existing knowledge and information society, the ability to process knowledge and information and to utilize computers has been emphasized. However, in a software-oriented society, problem solving ability that can solve new problems every day, higher-level competence such as critical thinking, communication, is required [1].

Computational thinking, which solves problems based on the principles of computer science, is emerging as an ability to solve various problems that arise in the software-oriented society. In addition, results of software education research based on computational thinking show that problem solving ability and high-level competence can be strengthened [2].

In developed countries such as the United States and the United Kingdom, software education is emphasized as a strategy to prepare for the fourth industrial revolution. For Korean elementary schools, the 2015 revised curriculum is designed to teach software courses over 17 hours a year [3]. Therefore, it is important to improve the software education capacity of pre-service teachers in order to strengthen software education at school.

Programming education is a representative education method that can strengthen software education capacity. It includes problem solving, procedural thinking process, algorithm production, programming implementation, and testing. Various researches are being conducted to improve computational thinking based on block programming education which can be easily accessed by novice users. However, beginners have difficulty in choosing a strategy for making program and procedural steps to find ways to solve problems. Also, it was found that the environment where learners can participate equally in the programming process influences learners' motivation and interest in learning [4, 5].

In recent years, studies have been conducted using visual thinking strategies to visually express ideas and concepts and to link them together to make it easier to understand the problem of learners [6]. The visual thinking strategy is useful for visualizing the learner's logical thought process so that it can be used to create procedures for problem solving or to utilize the information for making a program.

In this study, we developed a programming education model that uses visual thinking strategy to understand and solve problems presented in programming education for

beginners. This allows learners to visually create solutions to the problems presented on the basis of computational thinking elements. We also built a learning environment that allows learners to participate equally by using cloud-based collaborative document creation tools.

2. Background

A. Computational Thinking

Computational thinking means thinking that can solve everyday problems based on the theory and principles of computer science [7, 8]. In particular, the meaning of computational thinking in computer education can be summarized as follows [9].

First, new approaches and attempts are possible with regard to problem solving methods. Computational thinking is based on the principles of computer science. Therefore, it is possible to integrate various disciplines with factors such as problem analysis, problem decomposition, data collection, problem solving procedure, testing and error resolution, and generalization [10]. In addition, Bioinformatics is a discipline that can use a new level of problem-solving method by combining the principles of computational thinking and biology [11]. This method can be used to create new knowledge in combination with computational thinking in existing disciplines have.

Second, computational thinking is a way to think about the theories and principles of computer science. Computational thinking activities such as abstraction and modelling can be directly related to the structure of subjects in traditional computer science such as computer architecture. The theoretical-oriented view that is more fundamental than the engineering-oriented logic centered on practice suggests a high correlation with computational thinking power. It is highly related to theory and principle-centered views because computational thinking is ultimately a cognitive activity. Learning computational thinking ability can be a tool to approach theoretical computer science [9, 12].

Third, it is a tool to popularize computer science. According to Wing and Denning, the details of computational thinking are important concepts in computer science or an important tool and strategy for solving computing problems. However, many scholars, including Wing, have presented computational thinking as literacy skills for all citizens and students. Thus, learning of computing ability at the level of literacy ability can be an important means of learning computer science. In addition, computational thinking is a concept that encompasses not only technology but also cognitive science and humanities and social sciences, making it easier for ordinary citizens and students to access. It provides a pathway for computer science to become more popular [7].

In addition, the major elements involved in computational thinking are summarized as follows [13, 14, 15].

Table 1. Key elements of computational thinking

Area	Elements
Problem analysis	Problem understanding, Problem definition, Problem decomposition
Data analysis	Data collection, Data representation, Data structure
Abstraction	Pattern analysis, Logical reasoning, Modelling, Abstraction, Algorithm
Automation	Programming, Debugging, Automation
Generalization	Optimization, evaluation, application of case

B. Educational Programming Language

Unplugged computing, SW education using educational programming language, and education of physical computing have various SW education methods that can improve computational thinking. In particular, EPL (Educational Programming Language) such as Scratch, Entry, and code can be programmed by simply dragging and dropping commands composed of block shapes. In the case of Scratch program, EPL, which is widely used by students all over the world, can create various programs such as animation, storytelling, and games [16]. It also provides the ability to share programs with as many students as possible and re-mix and re-share the original source. The following is a list of studies using Scratch to improve computational thinking.

First, Karen Brennan and Mitchel Resnick used Scratch to create interactive stories to improve computational thinking and achieved meaningful results through design-based learning activities [17].

Second, Alex Ruthmann, Jesse M. Heines, Gena R. Greher, Paul Laidler, and Charles Saulters presented sound thinking to enable students to learn computational thinking using music live coding [18].

Third, Sze Yee Lye and Joyce Hwee Ling Koh studied learning methods that can support learner participation through evaluation of children's programming method, presentation contents and behavior in the Scratch [19].

Fourth, Maria José Marcelino, Teresa Pessoa, Celeste Vieira, Tatiana Salvador, and António José Mendesa developed a Scratch distance curriculum using Moodle for students' learning of computational thinking concepts [20].

Fifth, Jacob, S., Nguyen, H., Tofel-Grehl, C., Richardson, D., and Warschauer, M. used vocabulary cards to learn basic elements of the Scratch for people using English as a second language in computational thinking, and improved understanding of the programming environment [21].

In this way, programming learning using Scratch is effective in improving students' computational thinking. In addition, we can find solutions through abstracts of various

problems through computational thinking, and this ability can be a basic ability to solve fundamental problems arising in a complex society [22, 23].

C. Visual Thinking Strategy

The visual thinking strategy is applied to improve students' perception abilities in the field of education. Rudolf Arnhein noted that by recognizing what we see visually, our thoughts are combined when we interpret what we see [24]. This thinking strategy can enable learners to think creatively and critically in information processing, information analysis, conceptual understanding, and concept delivery that occur in the teaching and learning process [25]. Peter Copping has developed and applied a Visual Thinking strategy to improve computer programming for people with dyslexia, which has had a significant impact on improving the programming skills of students who are having trouble with text-based programming [26].

William L. Benzon suggested that visual thinking involves an imaginary and manipulative environment of ideas and is an important factor in human-machine interaction in computer science [27]. Therefore, the visual thinking strategy can visualize the steps that the learner has to perform procedurally in the problem solving method so that the beginner can easily access the visual learning strategy. An online mind map tool that utilizes Google Drive technology can visualize the logical process and ideas that learners make. The online collaborative learning environment allows learners to participate in equality [28].

3. Methodology

A. TREE (Thinking-Revise-Encoding-Evaluating) Model Design

Based on Younghoon Sung and Youngsik Jeong's research, a learning strategy consisting of TREE (Thinking-Revise-Encoding-Evaluating) was reused to ensure novice users to participate in the procedural thinking improvement and program implementation process [29].

First, the Thinking stage is the step of selecting a specific seed (Seed), which is a topic to be solved among various problems. Learner accesses the TREE system, finds various problems in problem pots and selects seeds to solve. The selected seeds can be viewed through related URLs via the URL associated with the Mind Map tool.

Second, the Revising phase is a step in creating a way to solve real problems from the selected seeds. To solve the problem of computational thinking based on the specific topic presented, learners break down the problem and produce decision-making procedures that can break down the problems, that is, refine the nodes and implement them in real programming. Through this process, the algorithms are specified by linking sprites and events required for Scratch. Learners can collaborate online using collaborative tools provided by the mind map program.

Table 2. Tree Model Design[29]

Step	Strategy	Student Activity	Tools
Thinking	Thinking seed from problem	● Connecting TREE system ● Seeking out seed from problem pots on TREE system ○ Pots contain many seeds that have Interesting code-projects ● Connecting Seed ○ Seed is a connection URL provided from Mind-map tool	TREE system
Revising	Revising nodes from seed	● Object programming by CT progression ● Decomposition from seed ○ Identified problems: seed ○ Make nodes from seed ● Finding Patterns ○ Adding or removing or adjusting nodes ○ Tied up similar nodes ● Defining algorithms ○ Categorize Sprite-Event-Script ○ Sort by Event ○ Abstraction Nodes	Collaborative Mind-map tools (MINDMUP 2.0 Program)
Encoding	Encoding node to code	● Matching blocks through the algorithms ● Stick blocks with script editor ● Building program BEAN: Beginner's Established Application from Nodes	Scratch Program
Evaluating	Evaluating project(codes)	● Testing BEAN: Beginner's Established Application from Nodes ● Debugging BEAN ● Sharing BEAN URL and Generalization	TREE system

Third, the Encoding step is a step of converting nodes created by the problem solving procedure into codes. The learners view the algorithm visualized through the online collaborative mind map and perform the Scratch programming by linking the related blocks. This creates BEAN, a program created by novices through nodes.

Fourth, the Evaluating step is the step of testing and evaluating the created program. Learners can share, share, modify, and evolve URLs using collaborative documents from the Google Cloud Platform.

B. Developing a Programming Curriculum

To apply the TREE model to the pre-service teachers, we developed a 15-week programming curriculum using Scratch. The curriculum structure and contents are as follows.

Table 3. Scratch Programming Curriculum based on TREE Model

Week	Contents
Week 1	Understand computational thinking Computing Principles Scratch basic
Week 2	(Understand) Action block (Utilization) Move sprite (Deepening) Sprite position control
Week 3	(Understand) control block (Utilization) 2 sprite control (Deepening) 2 Sprite position control and movement
Week 4	(Understand) control block - iteration (Utilization) Question and Answer Program (Deepening) My own Q & A program
Week 5	(Understanding) Motion Block, Type Block, Understanding Variables (Utilization) calculation program (Deepening) Sprite control by key input
Week 6	(Understand) pen block (Utilization) line drawing (Deepening) drawing polygon
Week 7~8	(Mission) Drawing a polygon with a pattern (Utilization) Simple Pattern Polygon (Deepening) drawing pattern polygon using random length
Week 9~10	Mission - Laser Shooting Game
Week 11	Create a small number decomposition program
Week 12~13	(Understand) data block (Use) Item Eating game (Deepening) Using list blocks Using additional blocks
Week 14	A variety of apple-shaped output programs
Week 15	Game Project

First, for the pre-service teachers who are learning to programme at the beginning level, the students are able to learn the concept of computational thinking and computing principles through simple examples and practices. The curriculum is designed to be able to program a combination of sprites, stages, and blocks.

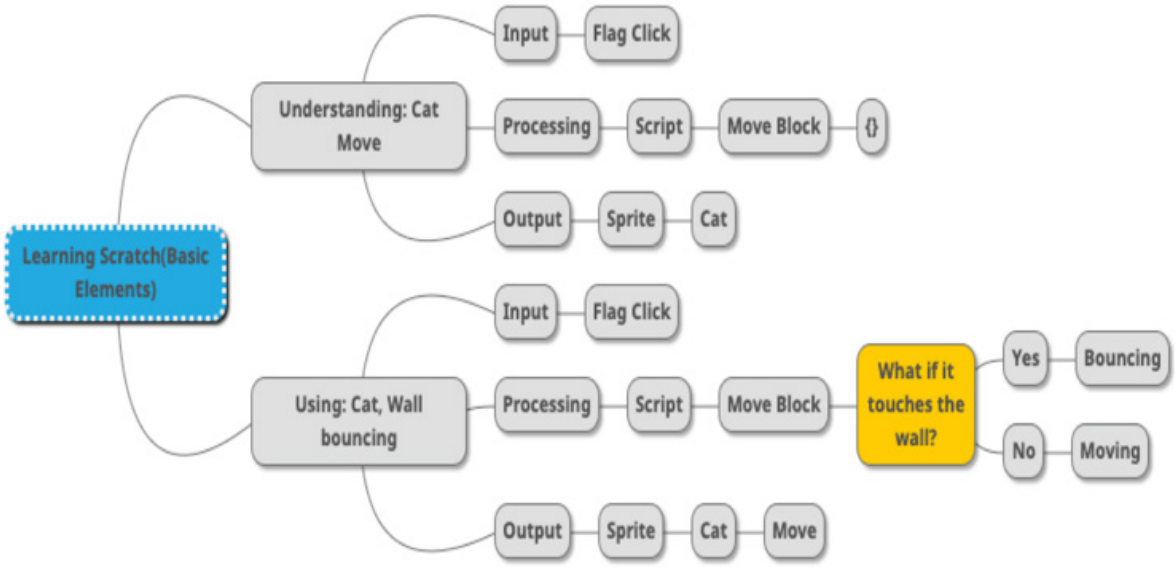

Figure 1. Visual Mind Map Expression Screen for Learning Scratch Basic Elements

As shown in <fig.1>, the components of Scratch are visualized so that learners who are learning to programme firstly can easily understand programming procedures and structures.

Second, it was operated for 2 ~ 8 weeks with 3 contents of understanding, utilization, and deepening in order to learn the scratch language. In the Understanding stage, in the comprehension phase, learners were able to learn basic blocks related to the problem.

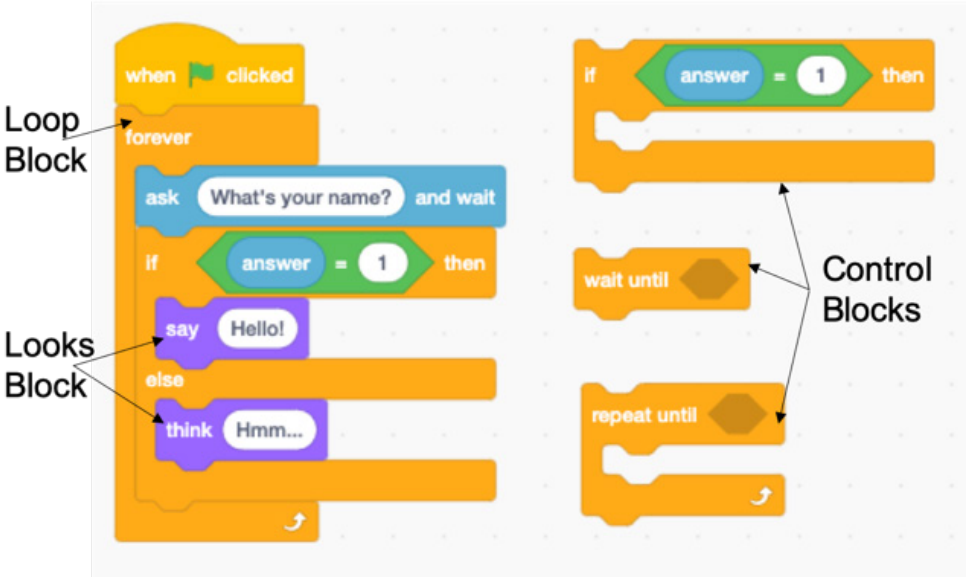

Figure 2. Understanding stage: Using basic blocks

As shown in <fig.2>, the learners looked at the basic blocks needed to make a simple calculation program and made it possible to construct them appropriately.

In particular, learning activities at the understanding stage correspond to the Thinking and Revising stage of the TREE Model.

As shown in <fig.3>, In order to express the problem solving process as a visual mind map, we applied the computing principle composed of input, processing, and output. This allows learners to visualize algorithms based on the nodes created in the mind map.

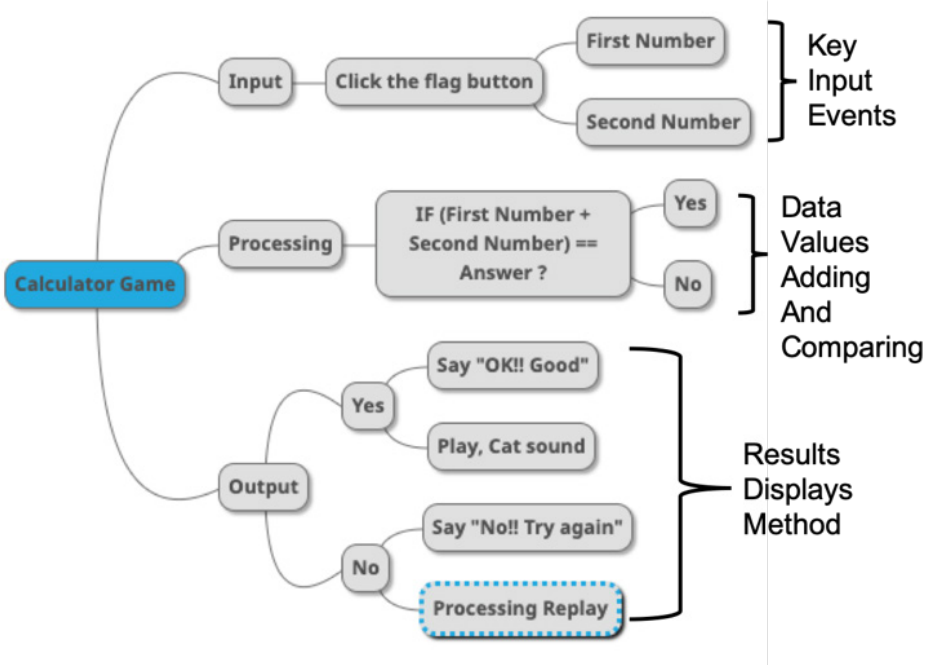

Figure 3. Connect with the Revising step to visualize based on Visual Thinking

In the utilization stage, we constructed a simple program using the learned blocks. This corresponds to the Encoding stage of the TREE Model. In the Revising stage, the algorithm expressed by the nodes of the mind map is coded by

Scratch block programming.

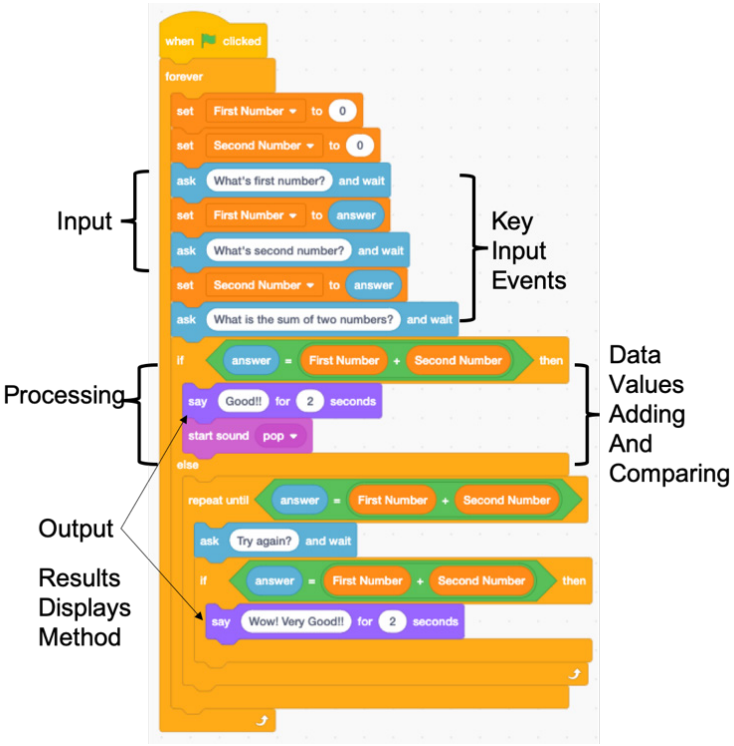

Figure 4. Utilization stage: View algorithms represented by nodes and create a simple program Connect

As shown in <Fig. 4>, the learner can convert and code into the scratch block programming according to the algorithm procedure expressed in nodes of the mind map like <Fig. 3>. Through this process, learners will create a simple program BEAN: Beginner's Established Application from Nodes.

In the deepening stage, learners have been able to create better programs at BEAN.

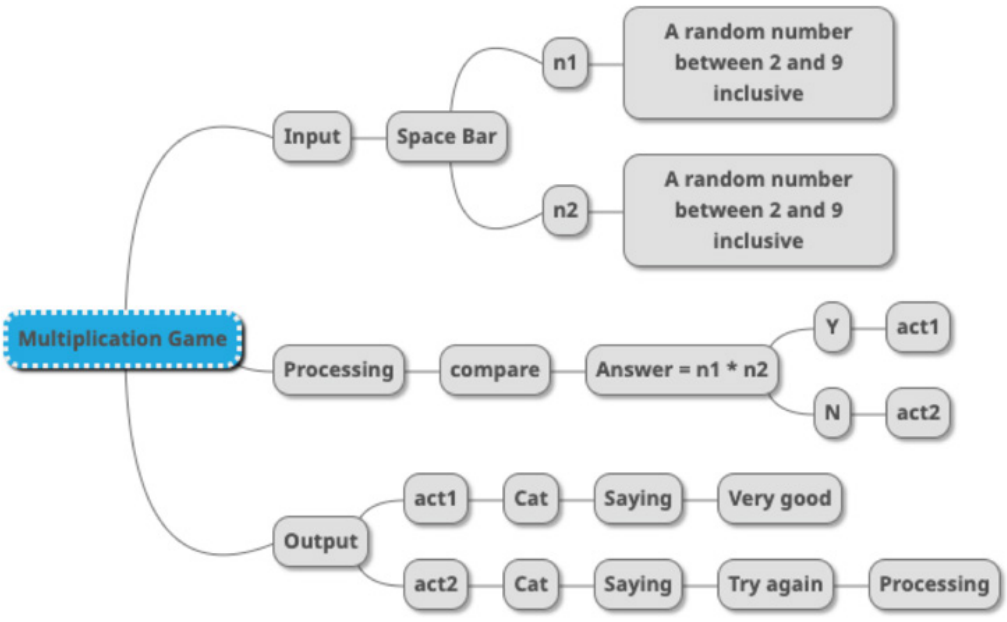

Figure 5. Deepening Steps: Visual Thinking Nodes (Algorithms) for Multiplication Game

<Fig. 5> is a visual representation of the algorithm proceeds as a visual thinking strategy for a learner to create a multiplication game program based on a simple calculation program.

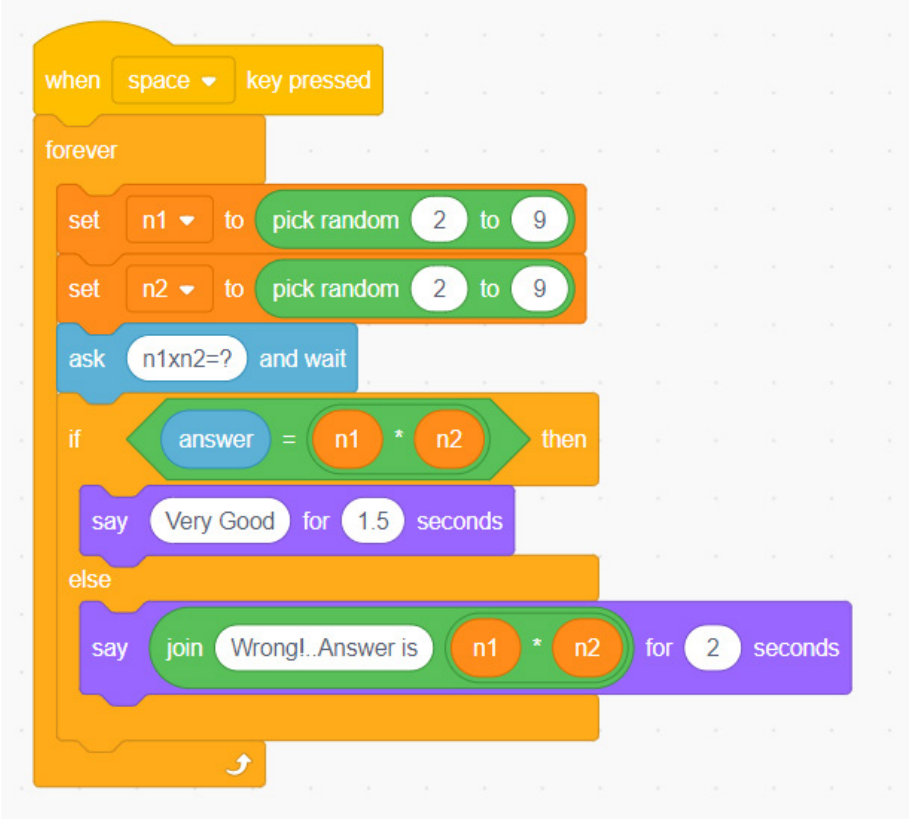

Figure 6. Scratch Block programming codes for multiplication games

<Fig. 6> is a screen that implements the Multiplication Game by looking at the algorithm which is a visually expressed node. Learners can use block learned in comprehension step to program block according to a visually expressed algorithm.

Third, it was constructed so that various programs can be created using the learned Scratch blocks until 9 ~ 14 weeks. The learners designed and developed their own programs according to the TREE Model process in order to solve the presented problems.

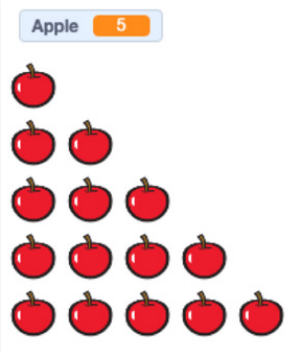

Figure 7. Create a variety of Scratch programs

As shown in <Fig.7>, the apple display program created by the learner is a program that is displayed with increasing apples according to the numerical value of the apple entered by the keyboard.

Fourth, in the 15th week, learners can create their own game programs based on the training they have learned for 1-14 weeks.

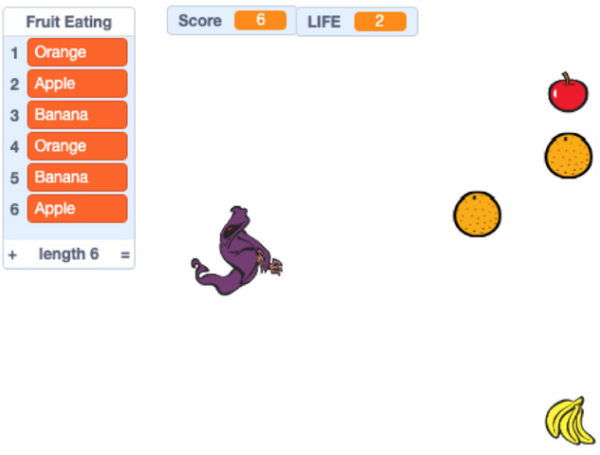

Figure 8. Implementing a Scratch game project

<Fig. 8> is a screen that implements the fruit-eating game with Scratch program.

4. Results and Discussion

A. Validation Design

This study was conducted for 15 weeks from March to June 2018 for students of software and programming language lectures among computer education pre-service teachers at J Education University in Korea. The results of this study are as follows: 22 male and 71 female respondents who answered the questionnaires were excluded.

The questionnaire consisted of 5 points scale and nominal scale questionnaire for the change of perception on computational thinking, block programming skill, interest and satisfaction of TREE model as in <Table 4> Respectively.

Table 4. Survey Questionaries

Div.	Questions
Tree Model	• Thinking: problem analysis, data analysis T1. Can you write down what you need to solve the problems presented? • Revising: Abstraction T2. Is it possible to structure and deploy the elements we have thought of at the computing stage (input, processing, output)? • Encoding: Automation T3. Can I view the structured elements according to computing procedures and implement them in block programming? • Evaluating: Generalization T4. Can I see the program links my friends have made and improve my program?
Computational Thinking Elements	C1. What is the most difficult part of computational thinking?
Interesting	I1. What is your programming interest?
Satisfaction	S1. What is your programming satisfaction?

B. Verification Result

As a result of the questionnaire response reliability analysis of the SW programming curriculum applying the TREE model applied in the study, Cronbach's α was found to be .806 and after .930, which satisfied 0.6 and above.

1) Effect of applying TREE model: As shown in <Table 5>, t-test results of SW education pre-test responding sample with TREE model showed that t-value was statistically significant as 3.640 (p = .000).

Table 5. The Comparison of TREE Model Effect between pre-test and post-test

Div.		M	SD	t	p
Total	Pre	2.718	.634	3.640	.000***
	Post	3.078	.763		
Thinking	Pre	2.527	.760	3.292	.001**
	Post	2.903	.848		
Revising	Pre	2.430	.826	3.215	.002**
	Post	2.817	.833		
Encoding	Pre	2.699	.831	2.044	.044*
	Post	2.946	.913		
Evaluating	Pre	3.215	.735	3.650	.000***
	Post	3.645	.880		

*p<.05, **p<.01, ***p<.001

In the TREE model, t-value is 3.292 (p = .001), t-value is 3.215 (p = .002) in Revising, t = 2.044 (p = .044) in Encoding and t = 2.044. The t-value was 3.650 (p = .000). The pre-post t-test showed statistically significant statistical results.

C. Learner's Block Programming Ability and Visual Thinking Strategy Effect

The t-test results showed that the t-value was 2.456 (p = .016), which was a significant predictor for the learners' block programming process.

Table 6. The Comparison of Learner's Programming Ability between pre-test and post-test

Div.		M	SD	t	p
Block programming Ability	Pre	2.817	.859	2.456	.016*
	Post	3.129	.900		

*p<.05

In addition, we examined the changes in perceptions of the most difficult computational thinking elements in the actual programming process in order to examine the change of the students' use of computational thinking elements through the use of Scratch blocks.

As shown in <Fig. 9>, we used the visual sinking strategy to select the most difficult elements among the computational thinking elements in the TREE model. As a result, learners were analyzed to find 'Finding Patterns' for problem solving procedures, and 'Programming' elements for programming and viewing mind map nodes made up of problem solving procedures.

It is analyzed that it is effective to make the problem solving procedures into visual nodes using the online mind map, which is a visual strategy tool used in the Revising stage of the TREE model and to transfer the mind map results to the programming. On the other hand, the abstraction or testing part has increased relatively, which seems to be caused by the increase in contents and level of curriculum.

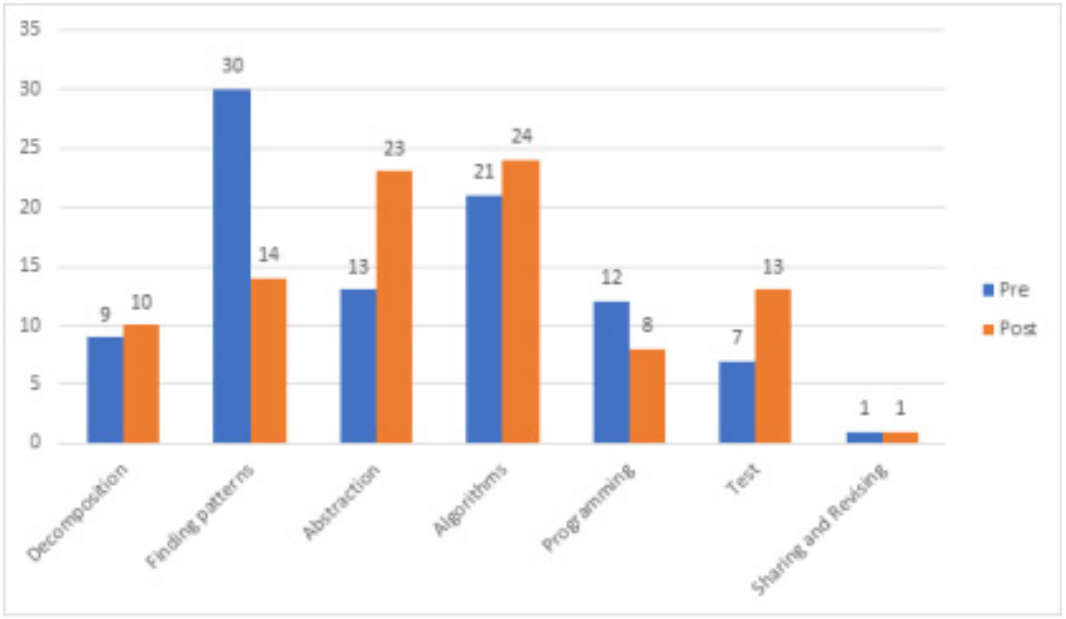

Figure 9. Changes in Difficulty Perceptions of Computational Thinking Elements Based on Visual Thinking Strategy

D. Interesting and Satisfaction

<Table 7> shows the change of interest and satisfaction of pre-service SW learners applying TREE model as follows.

Table 7. The comparison of Learners Interesting and Satisfaction between pre-test and post-test

Div.		M	SD	t	p
Interesting	Pre	2.946	.785	1.479	.143
	Post	3.140	1.028		
Satisfaction	Pre	2.968	.758	2.027	.046*
	Post	3.215	.883		

*p<.05

The t-test results showed that the t-value was 1.479 (p = .143). However, the t-value was 2.027 (p = .046), which was statistically significant.

The results of the analysis of the difference between men and women about interest and satisfaction of TREE model are as follows.

Table 8. The comparison of Gender Difference about Interesting and Satisfaction between pre-test and post-test

Div.		M	SD	t	p
Interesting	Female (N=71)	2.944	1.013	3.502	.001**
	Male (N=22)	3.773	0.813		
Satisfaction	Female (N=71)	3.113	0.919	2.043	.044*
	Male (N=22)	3.545	0.671		

*p<.05

As a result of the independent sample t-test, the t value was significant at 3.502 (p = .001) and the t value at the satisfaction level was 2.043 (p = .044). However, it was analyzed that there was a significant difference in the satisfaction of the programming curriculum using the TREE model.

5. Conclusions

In this study, we developed the TREE Model which is a programming education model that uses visual thinking strategy to understand and solve problems presented for pre-service teachers who are new to programming. The 15 - week curriculum was applied to the pre-service teachers and the results were as follows.

First, the effect of SW education on the pre-service teachers who are new to programming in the Thinking, Revise, Encoding, and Evaluating stages of the TREE model developed by the research is significant.

Second, applying the visual thinking strategy using the online-based mind map tool resulted in meaningful results in learners' block programming ability. In addition, the difficulty of the Finding Patterns element and the programming element that programmed according to the created procedure are reduced in the learner 's understanding of the most difficult part of the computational thinking element.

Third, applying the TREE model showed significant results in satisfaction. However, it seems that it is necessary to supplement the elements that can strengthen the learner 's immersion in the curriculum that changes the visual thinking logically in the interesting part and changes it to programming.

In order to generalize the research, it is necessary to apply the extension to the subjects.

REFERENCES

[1] Mahsa Mohaghegh, Michael McCauley, “Computational Thinking: The Skill Set of the 21st Century”, *International Journal of Computer Science and Information Technologies*. vol. 7, issue 3, pp. 1524-1530, Jun. 2016.

[2] Yu-Hui Ching, Yu-Chang Hsu, Sally Baldwin, “Developing Computational Thinking with Educational Technologies for Young Learners”, *TechTrends,* vol. 62, issue6, pp. 563-573, Nov. 2018.

[3] Namje Park, Younghoon Sung, Youngsik Jeong, Soo-Bum Shin, Chul Kim, “The Analysis of the Appropriateness of Information Education Curriculum Standard Model for Elementary School in Korea”*, Proc. International Conference on Computer and Information Science ICIS 2018: Computer and Information Science*, pp. 1-15, Sep. 2018.

[4] Jeong Ah Kim, Dae Young Ko, “Survey of On-Line & Block Programming Language-Scratch: On Perspective of Educational Achievements”, *CUTE 2017, CSA 2017: Advances in Computer Science and Ubiquitous Computing*, pp. 35-40, Dec. 2017.

[5] Younghoon Sung, “Development of SW Education Model based on HVC Learning Strategy for Improving Computational Thinking”, *Journal of The Korean Association of information Education*, vol.21, no.5, pp. 583-593, Oct. 2017.

[6] Po Ying Chu, Hsiu Yen Hung, Chih Fu Wu, Yen Te Liu, “Effects of various sketching tools on visual thinking in idea development”, *International Journal of Technology and Design Education*, vol. 27, issue 2, pp. 291–306, June 2017.

[7] Jeannette M. Wing, “Computational thinking”, *Magazine Communications of the ACM - Self managed systems*, vol. 49, issue 3, pp. 33-35, Mar. 2006.

[8] Joeng YoungSik, Soo-Bum Shin, Younghoon Sung, “A Study on the Relationship between Achievement Criteria and Computational Thinking in the Algorithms, Programming and Robotics and Computing”, *Journal of The Korean Association of information Education*, vol.21, no.1, pp. 105-114, 2017.

[9] Peter J. Denning, “Beyond Computational Thinking”, Communications of the ACM, vol. 52, no. 6, pp. 28-30, Jun. 2009.

[10] Cynthia C. Selby, "Computational thinking: The developing definition", *ITiCSE Conference*, University of Kent, Canterbury, England, July 1-3, 2013.

[11] Hong Qin, "Teaching computational thinking through bioinformatics to biology students", *Proceeding SIGCSE '09 Proceedings of the 40th ACM technical symposium on Computer science education*, pp. 188-191, Mar. 2009.

[12] Peter J. Denning, "Great principles in computing curricula", *Proceeding SIGCSE '04 Proceedings of the 35th SIGCSE technical symposium on Computer science education*, vol. 36, issue 1, pp. 336-341, Mar. 2004.

[13] Computing in the national curriculum: A guide for primary teachers (2013), Computing at School. Available: https://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

[14] Valerie Barr, Chris Stephenson, "Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?", Inroads, vol. 2, issue 1, pp. 48-54, Mar. 2011.

[15] Computer Science Fundamentals (2018). Code.org website. [Online]. Available: https://curriculum.code.org/csf-18/

[16] Scratch (2019). Lifelong Kindergarten Group: MIT Media Lab. Available: https://scratch.mit.edu/

[17] Karen Brennan, Mitchel Resnick, "New frameworks for studying and assessing the development of computational thinking", *In Proceedings of the 2012 annual meeting of the American Educational Research Association*, Vancouver, Canada, vol. 1, pp. 1-25, Apr. 2012.

[18] Alex Ruthmann, Jesse M. Heines, Gena R. Greher, Paul Laidler, Charles Saulters, "Teaching computational thinking through musical live coding in scratch", *Proceeding SIGCSE '10 Proceedings of the 41st ACM technical symposium on Computer science education*, pp.351-355, Mar. 2010.

[19] Sze Yee Lye, Joyce Hwee Ling Koh, "Case Studies of Elementary Children's Engagement in Computational Thinking Through Scratch Programming", *Computational Thinking in the STEM Disciplines*, pp. 227-251, Aug. 2018.

[20] Maria José Marcelino, Teresa Pessoa, Celeste Vieira, Tatiana Salvador, António José Mendesa, "Learning Computational Thinking and scratch at distance", Computers in Human Behavior, vol. 80, pp. 470-477, Mar. 2018.

[21] Jacob, S., Nguyen, H., Tofel-Grehl, C., Richardson, D., Warschauer, M., "Teaching computational thinking to English learners", *NYS TESOL journal*, vol. 5, issue 2, pp.12-24, Jan. 2018.

[22] Andrea A. Disessa (2000), Changing minds: Computers, learning and literacy, Cambridge, MA: MIT Press.

[23] (2019) Computational Thinking with Scratch website. [Online]. Available: http://scratched.gse.harvard.edu/ct/

[24] Rudolf Arnheim (2004). Visual Thinking Paperback – Deluxe Edition, University of California Press.

[25] Catherine McLoughlin, Krzysztof Krakowski, "Technological tools for visual thinking: What does the research tell us?", *Apple University Consortium Academic and Developers Conference*, Townsville, Queensland Australia, pp. 128-139, Sep. 2001.

[26] Peter Coppin, "Developing drawing and visual thinking strategies to enhance computer programming for people with dyslexia", *Proc. 2008 IEEE Symposium on Visual Languages and Human-Centric Computing*, pp.266-267, Sep. 2008.

[27] William L. Benzon. *Visual Thinking*. Encyclopedia of Computer Science and Technology. Volume 23, Supplement 8., CRC Press, 1990.

[28] (2018) Mindmup website, [Online], Available: https://www.mindmup.com

[29] Young-Hoon Sung, Young-Sik Jeong. "Design of the TREE Software Education Model Based on Visual Thinking for Primary Students". *Proc. International Conference on Future Information & Communication Engineering*. vol. 10, no. 1, pp. 45-47, Jun. 2018.