*International Journal of Engineering & Technology*, *7 (3.33) (2018) 46-50*

International Journal of Engineering & Technology

*Website: www.sciencepubco.com/index.php/IJET*

***Research paper***

# Prioritizing Learning Topics of Coding Curriculum for Elementary Students Using the Analytic Hierarchy Process

**Dongkyun Lim [1], Ji-Eun Lee[2], Dosik Moon[3], Gijun Um [4]**

*[1]Hanyang Cyber University, Seoul, Korea*
*[2]Hanyang Cyber University, Seoul, Korea*
*[3] Hanyang Cyber University, Seoul, Korea*
*[4] Hanyang Cyber University, Seoul, Korea*
**Corresponding author E-mail: eiger07@hycu.edu*

## Abstract

As the world confronts the 4th industrial revolution era, there is a growing interest in coding education around the world to cultivate creative and convergent students who possess computational thinking and problem-solving skills. In order for coding education to be successful, the following questions are considered: 1.What should be taught first? 2. How should it be taught? This study aims to determine the priority of leaning topics in elementary school coding education. To do so, a focus group interview was conducted with four experts in the field of coding education, and 12 learning topics were identified. Based on the interview results, a questionnaire was administered to coding instructors. The Analytic Hierarchy Process (AHP) was applied to derive priorities among the learning topics. The results showed that 'procedural problem solving' was found as the most important unit that the elementary school coding education needs to deal with. As for the learning topics, 'problem definition and breakdown', 'block coding', 'implementation of algorithm', 'understanding of algorithm' and 'necessity for learning coding' were found to be the top 5 priorities. Based on these results, this study presents four suggestions to consider for coding education to be carried out more effectively.

***Keywords***: *Coding education, curriculum development, computational thinking, the 4th industrial revolution, AHP*

## 1. Introduction

The 4th Industrial Revolution is reshaping every aspect of modern society, driven by rapid development in Information and Communication Technology (ICT). The industrial revolution paradigm has been creating dramatic shifts such as manufacturing to digital economies, and hardware-centered to software-centered approaches. With the advent of the digital economy, software is no longer confined to specific fields, but is being integrated into a variety of areas such as finance, economics, medicine, biology, and space engineering, creating various new jobs that require talented experts to solve complex and diverse problems that had never existed before. These days, having the ability to program is regarded as a critical skill to succeed in the Fourth Industrial Revolution era.
As the importance of software comes to the fore, many countries are trying to integrate coding into their basic education systems as a mandatory subject. For example, in Estonia, coding has been taught in public education since 1992, Israel since 2011, India since 2013, and in England, and Finland since 2014. In line with this global trend, the Korean government announced the 2015 Revised National Curriculum, and decided to make coding education mandatory starting 2018 in elementary and middle schools [1]. A compelling reason for promoting the teaching of coding in schools is not simply to cultivate future software developers or programmers but to foster creative and convergent students who possess computational thinking and problem-solving skills in preparation for the advent of more software-centered societies [2].

In order for coding education to be successful, it must have a detailed curriculum, qualified teachers, and a solid educational infrastructure. The role of teachers is particularly important for successful coding education, especially for younger students. Since many concepts of programming are abstract, it is difficult for elementary school students to learn these concepts effectively. Therefore, it is imperative to secure qualified teachers who have the skills necessary to effectively teach young learners about the contents, while also being able to properly teach computing thinking by supporting these students to overcome any educational difficulties. In Korea, however, a major challenge in successful implementation of coding education is the shortage of teachers with instructional competence. To solve these problems, it is necessary to develop training programs to enhance the capacity of the teachers.
This study was carried out with the aim of collecting basic data for the development of an effective instructor training course for elementary school coding education. As a first step to develop the training course, the following questions are considered: What should be taught first? and How it should be taught? To answer these questions, this study attempted to identify the priority of learning topics for elementary school coding education, and to decide what to teach for this level of students. In-depth interviews were conducted with experts in the field of coding education, and the Analytic Hierarchy Process (AHP) was applied to derive priorities among these learning topics.

## 2. Background

### 2.1. Coding and Computational Thinking

Code is a language in which information is conveyed, and coding refers to the process that the medium understands when information is transferred from one medium to another [3]. Coding education refers to teaching about coding, and it is also called SW education or programming education depending on researchers. Coding education has emerged in software-oriented societies where software is used mainly for the implementation and problem solving of ideas. Therefore, coding education is expected to cultivate a new type of computational thinking, which was not covered in traditional mathematics and science subjects [4].

Computational thinking refers to the ability to solve problems efficiently based on the understanding of fundamental computing concepts and principles [5]. Coding education is not limited to teaching how to deal with computers and smart devices, but it aims to develop computational thinking and problem solving skills through programming processes [6]. Although computational thinking is related to creating codes, but it is important to note that being able to write computer programming does not indicate computational thinking itself.

As Wing defined, computational thinking refers to an "intellectual process that allows problems to be established, and solutions are created to be performed effectively through a computing system" [7]. However, computational thinking is not just about computing. It is a fundamental ability to logically solve various problems in everyday life by actively utilizing computer resources as one does reading, writing, and calculating.

As Malyn-Smith et al. claimed, computational thinking involves skills that often include decomposition of a problem, pattern recognition, abstraction, and formulation of algorithms to solve similar problems or situations [8]. Therefore, coding education is related to improving creativity and problem-solving ability by enhancing computational thinking through programming. The ultimate goal of coding education is not to make a learner a program developer or a programmer, but to develop problem-solving skills through improving computational thinking. In the end, it is aimed to develop future talent capacity to solve various problems in each field.

### 2.2. Contents of Coding Curriculum

The goal of elementary school coding education in Korea is to "experience algorithms and programming based on sound information ethical consciousness to understand various problems in real life" [9]. To achieve this goal, it is important to determine what contents should be included in the curriculum to provide guidelines for coding teachers. For instance, in the 5th and 6th grade curricula, the coding education program consists of SW understanding, procedural problem solving, programming elements and structure. Also a majority of these programming activities are play-centered [10].

Finland, the benchmarking country for Korean coding education, conducts classes in three groups according to the level of the learners: From grades 1 to 2, classes are centered for learning the strategies and solving the problems; from grades 3 to 6, visual programming language are taught as an educational programming language and from grades 7 to 9, actual programming language are taught [11]. Compared with Korean educational contents, Finland's are more focused on problem solving and visual programming.

Given the successful implementation of Finland's coding curriculum, their approach could be adopted for the Korean education system. The contents of Korean elementary coding education also need to be designed to support students in developing logical thinking skills for real life problem solving. To do this, the curriculum of elementary school coding education should be structured to include the concepts of software, algorithms, programming, and problem solving. Also, the instructional design of the curriculum needs to be inquiry-based to provide students with opportunities to solve real life problems through exploring investigations, thinking critically, and testing solutions.

## 3. Methods

### 3.1. Research Procedures

In order to identify the core learning topics that should be incorporated in elementary school coding education, focus group interview was conducted with experts in the field (i.e. two professors of computer science and two coding instructors with at least five years of teaching experience in afterschool or private coding programs). Table 1 summarizes the general demographic characteristics of interviewees.

**Table 1:** Demographic characteristics of interviewees

| Interviewee | Gender | Age | Occupation | Teaching Experience |
|---|---|---|---|---|
| A | Male | 38 | Professor | 10 years |
| B | Male | 51 | Professor | 17 years |
| C | Female | 37 | Private school instructor | 7 years |
| D | Female | 42 | After-school instructor | 5 years |

The interview was guided by the following questions: 1. What are the perceptions of the students participating in coding education and the ability to be cultivated? 2. What should be taught in coding education? Based on the interview results, the researchers of this study developed a questionnaire that consisted of three upper hierarchy items of learning units and each unit had four lower hierarchy items of learning topics. Data was collected in a form of an open selection type and free write method. The questionnaire was distributed to 34 coding instructors who teach at private institutes and afterschool classes. At the end, 27 instructors responded.

### 3.2. Data Analysis

To determine the weight of each learning topic, the Analytic Hierarchy Process (AHP) a mathematical process designed to help define priorities and involves decision-making using both qualitative and quantitative variables, was employed [12]. AHP has basic principles including hierarchical structures, the relative priority of decision criteria, and consistent judgment. AHP builds a ranking of decision using comparisons, and criteria weights are automatically calculated. Based on the value of composite weight, the priority is derived.

When evaluating the relative importance across items, the analysis of reliability of the AHP was used by calculating the consistency ratio (C.R.). Generally, the consistency judgment can be proven when a lower value of C.R. is shown, and if this value is less than or equal to 0.1, the response can be considered to be relatively consistent [13]. The data collected through the questionnaire were analyzed after an exclusion of the samples that revealed a lack of reliability. Makeit, a commercial analysis tool for AHP, was used to determine the reliability and consistency of the AHP.

## 4. Results and Discussions

### 4.1. Summary of Focus Group Interview

All participants agreed that coding is a means to achieve the ultimate goal of coding education which is to develop logical and creative thinking. They believe that the primary role as coding teachers at the elementary school level is to help learners understand that coding is closely related to skills used in daily life and thus, it is necessary to keep the education engaging. The teachers also understand that computational thinking can be developed

through the process of planning and implementing algorithms while learners are involved in actual programming activities in class. Particularly, for young learners, block-based coding activities, in which Graphic User Interface (GUI) and logic can easily be learned, is much more useful than theory-based lectures that emphasize the memorization of already-written codes because block coding activities can attract learners' interest more easily. Therefore, the participating teachers recommend programming activities such as block coding to be incorporated in classes. Also, they emphasize that it is desirable to organize numerous group activities since these projects can enhance 'cooperative problem-solving ability'.
For the second question of 'What contents should be included in coding education?', the participants presented various opinions, which are categorized into three upper hierarchy items, i.e. major learning units: 'understanding of software', 'procedural problem solving', and 'programming elements and structure'. Each unit is subdivided into lower hierarchy items, i.e. learning topics and learning activities. Details of the learning topics and activities are summarized in Table 2.

**Table 2:** Learning topics and activities for coding education in elementary school

| Units | Learning topics | Learning activities |
|---|---|---|
| Understanding SW | Characteristics of an information society | Understanding characteristics of information society and the importance of ethics. |
| | Data and SW | Understanding the concept of data and the function of SW. |
| | Programming and coding | Understanding programming and coding concepts. |
| | Necessity for learning coding | Understanding the reasons for learning coding |
| Procedural problem solving | Problem definition and breakdown | Defining and breaking down problems. |
| | Design Thinking | Solving problems creatively through design thinking. |
| | Understanding algorithms | Understanding the concepts and principles of algorithms. |
| | Implementation of algorithms | Implementing algorithms using natural language and flowcharts. |
| Programming elements and structure | Block coding | Implementing algorithms using block coding. |
| | Coding English | Learning basic English for coding. |
| | Graphical thinking | Writing coding creatively through graphic thinking. |
| | Physical computing | Implementing algorithms using physical computing. |

Regarding the physical computing learning topic, the experts presented different views. Some argues that block coding rather than physical computing is more effective in understanding the concept of coding and thus believe it is better suited for elementary school education (Interviewee A and D), while others argued that physical computing, such as the SW used in a hamster robot, is more effective in attracting students (Interviewee B and C). Another interviewee also suggested that if block coding such as scratching was implanted, elementary English and design sense for coding would be helpful in enhancing the efficacy of the educational coding programs. (Interviewee D).

## 4.2. AHP Results

AHP was conducted to determine the priority of the learning topics presented in table 2. Table 3 presents the importance weights on learning units. The results of AHP showed that the highest priority among the units was 'procedural problem solving' (0.500) followed by 'programming elements and structure' (0.315), and 'understanding software' (0.185).

**Table 3:** Importance weights on learning units

| Units | W | C.R. | Learning topics |
|---|---|---|---|
| Understanding SW | 0.185 | 0.003 | Information society<br>Data and SW<br>Programming and coding<br>Necessity for learning coding |
| Procedural problem solving | 0.500 | | Problem definition and breakdown<br>Design Thinking<br>Understanding algorithms<br>Implementation of algorithms |
| Programming elements and structure | 0.315 | | Block coding<br>Coding English<br>Graphical thinking<br>Physical computing |

Table 4 presents the importance weights on learning topics in each unit and the overall importance weights of learning topics across all learning units. In each unit, the most important topics were found to be 'necessity for learning coding' (0.396), 'problem definition and breakdown' (0.376), and 'block coding' (0.479). Among sub-items of 'understanding SW', 'necessity for learning coding' (0.396) was of utmost priority, followed by 'data and SW' (0.309), 'programming and coding' (0.201), and 'information society' (0.093) Among sub-items of 'procedural problem solving', 'problem definition and breakdown' (0.376), was the most important topic, followed by implementation of algorithms' (0.296), 'understanding algorithms' (0.226), and 'design thinking ' (0.103). Among sub-items of 'programming elements and structure', 'block coding' (0.479) was of utmost priority, followed by 'coding English' (0.195), 'graphical thinking' (0.168), and 'physical computing' (0.158).

**Table 4:** Importance weights on learning topics

| Learning topics | W | OW | C.R. | Rank |
|---|---|---|---|---|
| Information society | 0.093 | 0.017 | | 12 |
| Data and SW | 0.309 | 0.057 | | 7 |
| Programming and coding | 0.201 | 0.011 | 0.037 | 11 |
| Necessity for learning Coding | 0.396 | 0.073 | | 5 |
| Problem definition and breakdown | 0.376 | 0.188 | | 1 |
| Design thinking | 0.103 | 0.051 | 0.113 | 9 |
| Understanding algorithms | 0.2260.296 | 0.008 | | 4 |
| Implementation of algorithms | | 0.148 | | 3 |
| Block coding | 0.479 | 0.151 | | 2 |
| Coding English | 0.195 | 0.053 | | 8 |
| Graphical thinking | 0.168 | 0.002 | 0.050 | 10 |
| Physical computing | 0.158 | 0.061 | | 6 |

(W=Weight, OW=Overall weight)

For overall weighted values across 12 learning topics, the coding teachers chose 'problem definition and breakdown' (0.188) as the top priority, followed by 'block coding' (0.151), 'implementation of algorithm' (0.148), 'understanding of algorithm' (0.113), and 'necessity for learning coding' (0.073). Fig. 1 presents the overall priorities across all the 12 items of learning topics.

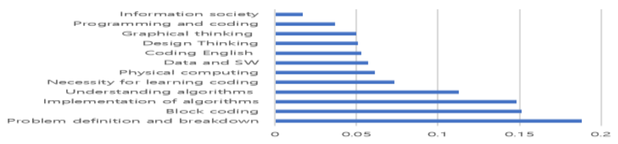

**Fig. 1**: Overall priorities across 12 learning topics

### 4.3. Pedagogical Implications

The results of the AHP analysis suggest to follow pedagogical implications for effective coding education in elementary school. Firstly, the curriculum in elementary school needs to be organized to develop students' logical thinking skills than programming skills. As shown evidently in the results, most learning topics related to ‘procedural problem solving’ are at the top of the priority list, compared to the ‘physical computing’ (0.061, 6th), which is one of the main activities in afterschool coding courses. This AHP analysis is consistent with the criticism that most of the current coding courses do not engage learners to be creative or to solve problems, but instead to copy codes that are already written or to code mechanically without sufficient understanding of how algorithms work.
Secondly, teachers for coding education need to come up with methods to supporting learners to have fun while being engaged in programming activities. Learners who are starting to learn coding from the beginning should have fun. Developing the ability to solve problems logically is as important as learning the syntax of a particular programming language. Thus, the first programming language to be taught should be carefully chosen, and the software environment for learning the programming language should be user-centered, including GUI. In addition, using multimedia-based coding examples or exercises can motivate students better than using text-based ones. Engaging in multimedia based activities, learners are expected to develop computational thinking and problem solving skills more easily.
Thirdly, coding instructors need to guide learners to understand the importance of learning to code. This learning topic is rarely dealt with in afterschool or private coding classes although it was found to be ranked relatively high (0.073, 5th) in this study. Many beginner level learners are reported to experience difficulties in learning to code and often lose interest due to the complexity of programming languages and teacher-centered pedagogical approaches. However, if students have a clear understanding of the necessity for learning to code, their achievement and satisfaction of the education can increase. In this respect, it is important to have a shared understanding between the instructor and the learners about the necessity of coding learning.
Lastly, coding curricula needs to be designed to converge diverse subjects. Some participants indicated that the basic knowledge of coding in English and graphical thinking can enhance the efficacy of code learning. For example, Scratch, a free downloadable block coding program, provides blocks and script to support learners' self-directed learning. As Fig 1 shows, all commands are presented in English which means students with a certain level of English are expected to use Scratch more easily than those who lack in English proficiency.

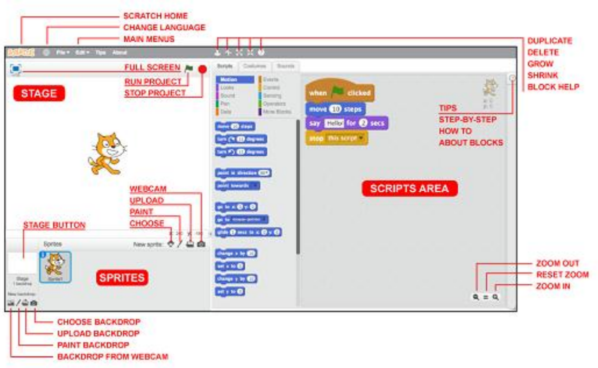

**Fig. 2**: Scratch script & blocks
<https://www.dummies.com/programming/opening-the-scratch-user-interface/>

Although coding education refers to programming education centered on algorithms and coding in a narrow sense, the eventual goal is to develop the application of high-order thinking skills acquired through coding education into solving a wide range of everyday problems.

## 5. Conclusion

Coding education has emerged as a new form of education for software-oriented society where software is used mainly for the implementation and problem solving of ideas. In Korea, coding education will be mandatory in elementary and middle school starting in 2018. In order to implement coding education successfully, it is necessary to prepare a curriculum that can effectively cultivate the computational thinking capacity of students in coding education. There is also a need to develop teacher training programs to provide teachers with information and guidance on what and how to teach appropriately.
This study was conducted as a part of the needs analysis for the development of a coding instructor training course. In order to determine the learning priorities of elementary school coding education, the researchers derived learning topics that should be dealt with in elementary school coding education, and presented the core contents through the priority of learning topics. The results showed that ‘procedural problem solving’ was found as the most important learning unit of the elementary school coding education. Among 12 learning topics identified to be covered in coding education, the top 5 learning topics were found to be ‘problem definition and breakdown’, ‘block coding’, ‘implementation of algorithm’, ‘understanding of algorithm’, and ‘necessity for learning coding’. These results are in line with existing research that the goal of coding education is to enhance computational thinking, i.e. logical thinking.
As the results of the study show, the basic objective of elementary school coding education is to develop logical thinking and problem-solving skills in learners and to help them find their own aptitudes for themselves. It is, therefore, crucial for students to understand the importance of coding education, for teachers to use logical procedures, and for the curricula to be designed to enhance student interest and achievement through diverse playful programming activities. In addition, it will be necessary to develop more effective pedagogical approaches that combine coding education in related areas such as design education and English education. The authors hope that these results will be used not only as a tool for designing actual curriculum for elementary school teachers, but also as basic data set for the development of further effective elementary coding education..

## Acknowledgement

This research was supported by KERIS (Korea Education & Research Information Service) and the Ministry of Education of Korea.

## References

[1] J. Ock and S. Ahn, “Analysis of Competency-Based In-service Training Programs for Informatics Teachers,” *The Journal of Korea Association of Computer Education,* Vol. 21, pp. 43-50, January 2018.
[2] H. Kim, Implications for Educational Informatization in Korea through OECD PISA 2012, Korea Education & Research Information Service, 2014.
[3] H. C, Von Baeyer, Information: The new language of science. Harvard University Press, 2004.
[4] H. J, Yun, “Performative Writing of Coding Game,” *Journal of Korea Game Society,* Vol. 16, No. 1, pp. 51-62, February 2016.
[5] Ministry of Education, SW Education Guideline, 2015.
[6] M. Mohaghegh and M. McCauley, “Computational Thinking: The Skill Set of the 21st Century,” *International Journal of Computer Science and Information Technologies,* Vol. 7 No. 3, pp.1524-1530, 2016.

[7] J. M. Wing, “Computational Thinking,” in Microsoft Asia Faculty Summit, pp. 1-59, Oct. 26, 2012.

[8] J. Malyn-Smith, B. Coulter, J, Denner, L. Lee, J. Stiles and L. Werner, “Computational Thinking in K-12: Defining the Space,” In Proc. 22th Society for Information Technology & Teacher Education International Conference, pp.3479-3484, March 2010.

[9] S. Kim, “Development of Scratch Code Analysis System for Assessment about Concepts of Computational Thinking,” *The Journal of Korea Association of Computer Education*, Vol. 18, No. 6, pp. 13-22, November 2015.

[10] H. J. Lee, “A Study on the Necessity of Convergence Approach in Coding Education through Case-study Interview with Individual Scratch learner,” *Korean Society of Basic design & Art,* Vol. 18, No. 6, pp. 487-500, December 2017.

[11] S. Shin. and Y. Bae, “Review of Software Education based on the Coding in Finland,” *Journal of The Korean Association of Information Education,* Vol. 19, No. 1, pp. 127-138, March 2015.

[12] U. Jung, S. Yim, S. Lim and C. Kim, “Bringing Kano`s perspective to AHP: The 2D-AHP decision model,” *Management and Production Engineering Review,* Vol. 7, No. 4, pp. 16-26, December 2015.

[13] J. Bentiez, X. Delgado-Galvan, J. Izquierdo and R. Perez-Garcia, “An Approach to AHP Decision in a Dynamic Context,*” Decision Support Systems,* Vol. 53, No. 3, pp. 499-506, June 2012.