Journal of The Korean Association
of Information Education Vol. 21, No. 1,
February 2017, pp. 105-114

http://dx.doi.org/10.14352/jkaie.2017.21.1.105

알고리즘, 프로그래밍, 로봇과 컴퓨팅 영역의 성취 기준과 컴퓨팅 사고력의 관련성 연구

정영식* · 신수범** · 성영훈***

전주교육대학교 컴퓨터교육과* · 공주교육대학교 컴퓨터교육과** ·
진주교육대학교 컴퓨터교육과***

요 약

소프트웨어 교육과 관련된 교육과정이나 운영지침에 포함된 Computational Thinking(CT)의 개념과 구성 요소가 서로 달라서 학교 현장에서 CT를 기반으로 한 소프트웨어 교육을 정착시키는 데 어려움을 겪고 있다. 따라서 본 연구에서는 전문가 델파이 조사 결과를 토대로, CT에 대한 용어를 '컴퓨팅 사고력'으로 통일하였고, CT의 구성 요소를 문제 정의, 자료 분석, 추상화, 자동화, 일반화 등 5가지로 구분하였다. 또한, 한국정보교육학회에서 개발한 정보과 교육과정 표준 모델에서 컴퓨팅 사고력과 관련성이 높은 소프트웨어 영역을 선정하여, 하위 영역별 성취 기준과 관련성이 높은 컴퓨팅 사고력이 무엇인지를 조사하여 제시하였다.

키워드 : 알고리즘, 프로그래밍, 로봇과 컴퓨팅, 성취 기준, 컴퓨팅 사고력, 정보과 교육과정

A Study of the Connection between Achievement Criteria and Computational Thinking in the Areas of Algorithms, Programming and Robotics, and Computing

Youngsik Jeong* · Soobum Shin** · Younghoon Sung***

Jeonju National Univ. of Education* · Gongju National Univ. of Education** ·
Jinju National Univ. of Education***

ABSTRACT

Because the concepts and components of computational thinking included in the Information Education Curriculum and the Software Education Guidelines are different, it has been difficult to establish computational thinking-based software education in schools. Therefore, this study, which is based on the Delphi survey results from 39 experts, we defined computational thinking as 'computing thinking' and separated the components of computational thinking into five main categories: (1) problem definition, (2) data analysis, (3) abstraction, (4) automation, and (5) generalization. In addition, we selected software areas that are strongly related to computational thinking in the KAIE's information Curriculum Standard Model and surveyed experts to decide which computing

교신저자 : 성영훈(진주교육대학교, yhsung@cue.ac.kr)
논문투고 : 2016-12-00
논문심사 : 2017-00-00
심사완료 : 2017-00-00

thinking components are related to the achievement criteria of the software areas.

Keywords : algorithm, programming, robotics and computing, achivement criteria, computational thinking, information education curriculum.

1. 연구의 필요성 및 목적

교육부는 2015 개정 교육과정에서 소프트웨어 교육을 강화하기 위해 초등학교 5~6학년을 대상으로 한 실과 교육과정에 소프트웨어의 이해, 절차적 문제 해결, 프로그래밍 요소와 구조 등을 포함하였으며, 그동안 중학교 선택 교과이었던 '정보'를 별도로 '정보과'로 독립시켜 필수 교과로 편성하였다[13].

초중등학교에서의 소프트웨어 교육의 목적은 프로그래머를 키우는 것이 아니라 Computaional Thinking(이하 CT)을 키우는 데 있다[19]. 그러나 CT에 대한 용어와 개념은 교육부가 고시한 초중등학교 교육과정 내에서도 다르게 표현하고 있다. 2009 개정 교육과정의 '정보' 과목에서는 CT를 '계산적 사고'로 번역하고, 그 개념을 '정보 과학 기술의 기본 개념과 원리를 이해하고, 실생활의 다양한 문제를 관찰하고 해결하는 능력'으로 정의하였고[14], 2015 개정 정보과 교육과정에서는 CT를 '컴퓨팅 사고력'으로 번역하고, 그 개념을 '컴퓨터과학의 기본 개념과 원리 및 컴퓨팅 시스템을 활용하여 실생활과 다양한 학문 분야의 문제를 이해하고 창의적으로 해법을 구현하여 적용할 수 있는 능력'으로 정의하고 있다[13]. 그 외의 많은 학자들은 CT를 컴퓨터적 사고력[16], 컴퓨터 과학적 사고[19], 정보적 사고[8], 정보과학적 사고[17] 등 다양한 표현을 사용하고 있다.

또한, CT의 하위 개념을 2015 개정 정보과 교육과정이나 2015 소프트웨어 교육 운영 지침에서 서로 다르게 표현하고 있다. 2015 소프트웨어 교육 운영 지침에서는 CT를 컴퓨팅 사고력으로 정의하고, 그 하위 구성 요소를 구조화, 조직화, 추상화, 자동화, 최적화, 일반화 등 6가지로 구분하고 있지만[12], 소프트웨어 교육 운영 지침 개발 연구에 나온 해설서에서는 컴퓨팅 사고력의 구성 요소를 크게 추상화와 자동화로 재구분하고 있다[18]. 이것은 이후에 발표된 2015 개정 정보과 교육과정에도 반영되어 컴퓨팅 사고력의 구성 요소를 추상화(abstraction) 능력과 자동화(automation) 능력으로 구분하게 만들었다.

영국이나 미국과 같은 주요 선진국에서는 특정 교과 영역이 아닌 다양한 교과 영역에서 CT를 교과 수업 내용과 통합하려 하고 있으나, 2015 정보과 개정 교육과정에서는 특정 교과의 특정 단원으로만 한정하고 있어 CT 개념을 폭넓게 적용하지 못하고 있다[9].

이러한 CT 용어에 대한 혼란, 불분명한 하위 구성 요소, 교육과정 성취 기준과의 비연결성 등으로 인해 CT를 목적으로 한 소프트웨어 교육이 학교 현장에 정착하는 데 어려움을 겪고 있다. 따라서 본 연구에서는 이러한 문제를 해결하고자 CT 용어에 대한 합의된 용어 정의, CT의 하위 구성 요소에 대한 정리, 교육과정 성취 기준에 따른 CT의 관련성을 조사하였다. 특히 소프트웨어 교육 영역 중에서 CT와 가장 관련성이 높은 알고리즘, 프로그래밍, 로봇과 컴퓨팅 등 3개 영역의 성취 기준과 CT의 하위 개념을 연결함으로써 교육 목적에 부합하는 소프트웨어 교육이 될 수 있도록 기초 자료를 제공하였다.

2. 연구 내용 및 방법

본 연구의 내용은 CT에 대한 적절한 용어와 하위 구성 요소를 선정한 후, 알고리즘, 프로그래밍, 로봇과 컴퓨팅 등 소프트웨어 교육의 성취 기준과 관련된 적절한 CT의 하위 개념을 선정하였다.

이러한 연구 문제를 해결하기 위해 정보교육 관련 전문가를 대상으로 델파이 조사를 실시하였다. CT의 용어나 하위 개념, 성취 기준과의 관련성은 명확한 답이 있기보다는 학자마다 주장하는 바가 다를 수 있으므로, 델파이 조사와 설문조사를 통해 합의를 도출하였다. 용어

와 하위 개념에 대한 델파이조사에는 39명이 참여하였고, 성취 기준과의 관련성에서는 19명의 전문가가 설문에 참여하였다.

CT 용어와 하위 구성 요소의 적절성을 평가하기 위해 리커드 척도를 활용하였으며, 전혀 적절하지 않다는 1점, 적절하지 않다 2점, 보통이다 3점, 적절하다 4점, 매우 적절하다 5점 등으로 수정하였다. 또한, 조사 결과의 타당도를 분석하기 위해 타당도 비율(CVR; Content Validity Ratio)을 계산한 후 패널 수가 39명임에 따라 CVR 값이 0.33 이상인 경우에만 타당한 것으로 판단하였다[9].

$$CVR = \frac{N_e - \frac{N}{2}}{\frac{N}{2}}$$

N_e: 적절하다와 매우 적절하다는 응답자 수

N : 전체 응답자 수

3. CT 용어에 대한 합의

CT는 Newell 외 2명(1960)이 제시한 알고리즘적 사고에서 유래되었으며[7], Wing에 의해 학생들의 필수 역량으로 자리잡고 그 개념이 세분화되고 발전하기 시작하였다[11].

첫째, Wing은 정보를 처리하는 인간이나 컴퓨터가 해결책을 효과적으로 수행할 수 있는 형태로 표현되도록 문제와 그 해결책을 고안하는 사고 과정이라고 정의하였으며[10], Brigitte는 사람이 혼자서는 할 수 없는 일을 성취하는 것으로서 인간 지능의 한계와 기계의 능력에 대해 이해하고 문제를 해결하는 것이라 정의하였다[3]. Quick Start Computing은 컴퓨터로 작업을 시작하기 전에 착수하는 생각으로서, 컴퓨터가 우리를 도와주는 방식으로 문제 해결 방법을 이끌어내는 과정이나 접근법이라 하였고[15], Barefoot Computing은 컴퓨터의 도움이 있거나 없을 때 문제를 효과적으로 해결할 수 있도록 도와주는 능력과 기술을 발전시키는 것이라 정의하였다[1].

이들의 개념을 종합해보면, CT는 컴퓨터나 기계가 문제를 해결할 수 있도록 준비하는 과정과 컴퓨터가 처리하는 방식대로 문제를 해결하는 과정으로 구분할 수 있다.

첫째, 준비하는 과정이란 문제를 찾아 정의하고, 컴퓨터나 기계가 해결할 수 있는 문제인지 판단하고, 만약 그렇다면 컴퓨터나 기계가 문제를 해결할 수 있도록 절차를 만드는 것을 의미한다.

둘째, 컴퓨터가 처리하는 방식이란 컴퓨터처럼 사고하는 것을 의미하는 것은 아니라, 컴퓨터가 처리하는 방식대로 문제를 해결한다는 것을 의미한다.

따라서 CT의 개념을 컴퓨터과학의 개념과 원리를 이해하고 그것을 기반으로 문제를 해결하는 능력이라고 정의할 수 있다.

이러한 CT 개념에 대한 1차 델파이 조사 결과, 전문가들은 컴퓨팅 사고(력), 컴퓨터적 사고(력), 컴퓨터 과학적 사고(력), 정보 사고(력), 정보적 사고(력), 정보 과학적 사고(력), 계산적 사고(력), 전산적 사고(력) 등 다양한 표현을 제시하였다. 이에 대해 2차 델파이 조사를 실시한 결과, <Table 1>과 같이 컴퓨팅 사고(력)에 대한 긍정률이 73.8%로 가장 높게 나타났다. 다음으로는 정보 사고(력)이 41.5%로 높게 나타났고, 컴퓨터 과학적 사고(력)은 35.7%, 컴퓨터적 사고(력)은 33.3%를 나타내었다. 또한, 전문가 의견에 대한 타당도(CVR)는 39명이 참여하였기 때문에 그 값이 0.33 이상이면 타당한 것으로 판단할 수 있는데, 컴퓨팅 사고(력)의 CVR 값이 0.48로 유일하게 타당한 것으로 나타났다.

<Table 1> Terminology for CT

Terminology	Mean	Median	Q1~Q3	Positive	CVR
컴퓨팅 사고(력)	3.90	4.0	3.3~5.0	73.8	0.48
컴퓨터적 사고(력)	2.86	3.0	2.0~4.0	33.3	-0.33
컴퓨터 과학적 사고(력)	3.17	3.0	3.0~4.0	35.7	-0.29
정보 사고(력)	3.00	3.0	2.0~4.0	41.5	-0.17
정보적 사고(력)	2.31	2.0	2.0~3.0	9.5	-0.81
정보 과학적 사고(력)	2.71	3.0	2.0~4.0	28.6	-0.43
계산적 사고(력)	2.36	2.0	1.3~3.0	19.0	-0.62
전산적 사고(력)	1.88	2.0	1.0~2.0	4.8	-0.90

CT에 대한 상위 3개의 개념만을 대상으로 2차 델파이조사를 실시한 결과 <Table 2>와 같이 1차와 마찬가지로 컴퓨팅 사고(력)이 81.3%로 가장 높게 나타났으며, 컴퓨팅 사고(력)의 CVR 값이 0.63으로 유일하게 타당한

것으로 나타났다.

<Table 2> Terminology for CT

Terminology	Mean	Median	Q1~Q3	Positive	CVR
컴퓨팅 사고(력)	4.25	4.5	4.0~5.0	81.3	0.63
컴퓨터 과학적 사고(력)	3.81	4.0	3.0~4.0	62.5	0.25
정보 사고(력)	3.50	4.0	3.0~4.0	56.3	0.13

컴퓨팅 사고력에 대한 용어가 이미 2015 개정 교육과정에서 명시하고 있으므로 전문가들이 CT를 컴퓨팅 사고력으로 해석하는 것을 동의한 것으로 판단된다. 따라서 본 논문에서도 이후부터는 CT를 컴퓨팅 사고력으로 표기하였다.

4. 컴퓨팅 사고력의 하위 구성 요소

2015 개정 정보과 교육과정에서 컴퓨팅 사고력의 개념을 추상화와 자동화로 구분하였다. 추상화는 문제의 복잡성을 제거하기 위해 사용하는 기법으로 핵심 요소 추출, 문제 분해, 모델링, 분류, 일반화 등으로 구분하고 있으며, 자동화는 추상화 과정을 통해 도출된 문제 해결 모델을 프로그래밍하는 것으로 설명하고 있다. 즉, 추상화와 자동화를 컴퓨팅 사고력의 핵심 요소로 표현하고 있다는 것이고, 차이점은 세부 구성 요소인 일반화를 소프트웨어 교육 운영 지침에서는 자동화에 포함시키고 있으나, 정보과 교육과정에서는 추상화에 포함하고 있다[13][12]. 또한, 소프트웨어 교육 운영 지침 개발 연구에서 하위 영역인 추상화와 상위 개념인 추상화가 이중으로 표현되어 있어 교사들에게 정확한 개념을 이해시키는 데 어려움을 주고 있다[18].

컴퓨팅 사고력은 학자마다 다른 구성 요소를 포함하고 있으며, 이들의 주장을 정리한 결과 <Table 3>과 같이 문제 분해, 자료 수집, 자료 분석, 자료 표현, 패턴 인식, 논리적 추론, 추상화, 모델링, 알고리즘, 자동화, 시뮬레이션, 병렬화, 일반화, 평가 등 14가지 하위 요소로 구분할 수 있었다[1][2][3][6][10][15].

1차 델파이 조사 결과를 반영하여, 문제 분해를 문제 이해와 문제 정의, 문제 분해로 구분하였고, 시뮬레이션과 병렬화는 자동화로 통합하였다. 또한, 프로그램의 오류를 수정하는 디버깅과, 프로그램의 성능을 개선하는 최적화를 새롭게 추가하여 다음과 같이 16가지 요소로 구분하였다.

<Table 3> Components of CT

	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	⑪	⑫	⑬	⑭
CAS	0						0		0				0	0
CSTA & ISTE		0	0	0			0		0	0	0	0		
CTillusted	0		0		0		0	0	0				0	
Wing							0			0				
code.org	0				0		0		0					
QuickStar					0	0	0		0					0
Barefoot	0				0	0	0		0					0

① problem decomposition, ② data Collection ③ data Analysis ④ data representation, ⑤ pattern recognition, ⑥ logical reasoning, ⑦ abstraction, ⑧ modeling, ⑨ algorithmic, ⑩ automation, ⑪ simulation, ⑫ parallelization, ⑬ generalisation, ⑭ evaluation

- 문제 이해: 문제의 현재상태와 목표 상태를 알수알 수 있다.
- 문제 정의: 문제 상황을 파악하여 문제를 표현할 수 있다.
- 문제 분해 : 복잡한 문제를 쪼개어 생각할 수 있다.
- 자료 수집: 문제해결에 필요한 자료를 수집할 수 있다.
- 자료 분석: 문제 해결을 위해 자료를 분석할 수 있다.
- 자료 표현: 분석된 결과를 말, 글, 그림 등으로 표현할 수 있다.
- 패턴 분석: 문제해결 과정에서 반복되는 요소를 찾을 수 있다.
- 논리적 추론: 알고리즘과 프로그램의 결과를 예측할 수 있다.
- 모델링: 문제해결하는 데 불필요한 요소를 제거할 수 있다.
- 알고리즘: 문제해결과정을 그림이나 순서도, 의사코드로 나타낼 수 있다.
- 프로그래밍: 프로그램을 작성하여 문제를 해결할 수 있다.
- 디버깅: 프로그램의 오류를 찾아 수정할 수 있다.
- 추상화: 문제해결에 필요한 절차와 방법을 단순화하여 나타낼 수 있다.
- 자동화: 알고리즘을 만들고 프로그래밍을 통해 문

제를 해결할 수 있다.

- 최적화: 보다 나은 문제해결 과정으로 개선할 수 있다.
- 일반화: 문제 해결 과정을 유사한 문제에 적용할 수 있다.

컴퓨팅 사고력의 하위 요소에 대한 용어 정의가 적절한지 전문가를 대상으로 한 델파이 조사를 실시한 결과 <Table 4>와 같이 적절하다고 긍정적으로 응답한 비율이 모두 높게 나타났으며, CVR의 타당도 수치도 최소 0.50보다 높아서 하위 요소에 대한 개념이 모두 적절하게 정의되었다고 응답하였다.

<Table 4> Definition for the Components of CT

Components of CT	Mean	Median	Q1~Q3	Positive	CVR
① problem understanding	4.13	4.0	4.0~5.0	81.3	0.63
② problem definition	3.94	4.0	4.0~4.0	81.3	0.63
③ problem decomposition	4.50	5.0	4.0~5.0	87.5	0.75
④ data Collection	4.19	4.0	4.0~5.0	81.3	0.63
⑤ data Analysis	4.50	5.0	4.0~5.0	93.8	0.88
⑥ data representation	4.50	5.0	4.0~5.0	87.5	0.75
⑦ pattern recognition	4.44	4.5	4.0~5.0	93.8	0.88
⑧ logical reasoning	4.06	4.0	3.8~5.0	75.0	0.50
⑨ modeling	3.94	4.0	3.8~4.0	75.0	0.50
⑩ algorithmic	4.56	5.0	4.0~5.0	87.5	0.75
⑪ Programming	4.31	4.0	4.0~5.0	87.5	0.75
⑫ Debuging	4.00	4.0	3.8~4.3	75.0	0.50
⑬ abstraction,	4.44	4.5	4.0~5.0	93.8	0.88
⑭ automation,	4.25	4.0	4.0~5.0	81.3	0.63
⑮ optimization	4.13	4.0	4.0~4.3	87.5	0.75
⑯ generalisation	4.13	4.0	4.0~4.0	93.8	0.88

컴퓨팅 사고력의 하위 요소가 너무 많아서 다음과 같이 하위 요소를 문제 정의, 자료 분석, 추상화, 자동화, 일반화 등 5가지로 그룹핑하였다.

- 문제 정의: 해결할 문제를 이해하고, 표현하고, 분해하는 과정이다. 문제가 무엇인지 인식하고, 그 문제의 현재 상태와 목표 상태를 파악하여 문제를 표현한다.
- 자료 분석: 문제 해결을 위해 필요한 자료를 수집하고, 분류하고, 구조화하는 과정이다. 수집된 자료는 분석 과정을 통해 말과 글, 그림을 통해 표현되고, 문제 해결에 도움을 줄 수 있도록 표나 그래프 등으로 구조화할 수 있다.
- 추상화: 문제와 자료를 분석하여 패턴을 찾고, 논리적 추론과 모델링을 통해 문제 해결 방법을 그림이나 순서도, 의사코드와 같은 알고리즘으로 표현하는 과정이다.
- 자동화: 추상화된 문제 해결 과정을 실제로 프로그래밍을 통해 구현하고, 오류를 찾아 수정하는 과정을 의미한다.
- 일반화: 문제 해결 과정이 옳은지, 충분히 빠른지, 자원 사용에 있어서 경제적인지, 사람들이 적절한 경험을 이용하고 촉진하기 쉬운지 등을 평가하고, 문제점을 개선하여 보다 나은 해결 방법을 찾아 유사한 문제에 적용하는 과정이다.

5. 소프트웨어 성취 기준과 컴퓨팅 사고력

한국정보교육학회에서는 초중등 소프트웨어 교육을 위해 '정보과 교육과정 표준 모델'을 개발하였고, 내용 영역을 소프트웨어, 컴퓨팅시스템, 정보생활 등 3개로 구분하였다[4]. 이 중에서 컴퓨팅 사고력과 연관성이 높은 소프트웨어 영역의 성취 기준을 중심으로 컴퓨팅 사고력과의 관련성을 전문가 19명을 대상으로 설문조사를 실시하였다.

정보과 교육과정 표준 모델에 제시된 소프트웨어 영역은 알고리즘, 프로그래밍, 로봇과 컴퓨팅 등 3개 세부 영역으로 구분되어 있다[4]. 따라서 소프트웨어 영역의 성취 기준과 관련된 컴퓨팅 사고력을 조사하기 위해 알고리즘, 프로그래밍, 로봇과 컴퓨팅 등 각 영역별 성취 기준과 해설을 제시하고, 각각의 성취 기준에 가장 적합한 컴퓨팅 사고력의 하위 요소를 선택하도록 하였다.

5.1 알고리즘

알고리즘 영역은 정보기기를 이용하여 생활 속 문제를 해결할 수 있도록 문제에 대한 본질적인 이해와 분석, 일을 처리하기 위한 논리적 절차의 흐름에 대해서 알아보고, 정보기기가 이해할 수 있는 프로그램으로 적용하기 위한 내용으로 구성하였다.

Achievement criteria of algorithm	Pr	Da	Ab	Au	Ge
Understanding the work sequence[Pr]	89.5	5.3	5.3	0.0	0.0
Representation of the work sequence[Da]	21.1	63.2	15.8	0.0	0.0
Algorithm definition[Ab]	31.6	5.3	63.2	0.0	0.0
The relevance of algorithms and program [Au]	5.3	15.8	26.3	52.6	0.0
Expressing the problem-solving process in words and in writing[Da]	10.5	63.2	15.8	0.0	10.5
Represent the problem-solving process with pictures and symbols[Ab]	0.0	42.1	47.4	5.3	5.3
Finding and representing patterns[Ab]	0.0	31.6	68.4	0.0	0.0
Expression of conditional solution[Ab]	0.0	5.3	47.4	31.6	15.8
Flowchart and pseudo-code[Ab]	0.0	15.8	68.4	15.8	0.0
The relevance of algorithm and decomposition[Pr]	36.8	31.6	21.1	5.3	5.3
Expressing various flowcharts[Ab]	0.0	0.0	84.2	5.3	10.5
Describe the current and target status of the problem[Pr]	68.4	15.8	15.8	0.0	0.0
Algorithm improvements[Au]	0.0	5.3	21.1	47.4	26.3
Understanding search algorithm[Ab]	26.3	5.3	36.8	21.1	10.5
Understanding sorting algorithms[Ab]	21.1	5.3	42.1	21.1	10.5
Algorithm analysis[Da, Ge]	5.3	26.3	21.1	21.1	26.3
Algorithm Evaluation[Ge]	0.0	5.3	26.3	21.1	47.4
Application of Optimal Algorithm[Ge]	5.3	0.0	5.3	15.8	73.7

(Fig. 1) Algorithm and computing thinking

※ Pr: Problem Definition, Da: Data analysis, Ab: Abstraction, Au: Automation, Ge: Generalisation

알고리즘 영역의 성취 기준과 관련된 컴퓨팅 사고력의 하위 요소를 설문한 결과 (Fig. 1)과 같이 분석되었다. 컴퓨팅 사고력과의 관련성이 높다고 응답한 비율이 최대값인 구성 요소를 우선적으로 선정하였다. 그러나 최대값과 차이가 5% 이내인 차순위 구성 요소가 있다면 그것을 함께 병기하였다.

- 일의 순서 이해[Pr] : 일에는 순서가 있음을 이해할 수 있다.
- 일의 순서 표현[Da] : 일의 순서에 따라 그림이나 문장을 나열할 수 있다.
- 알고리즘 의미[Ab] : 알고리즘의 의미를 이해할 수 있다.
- 알고리즘과 프로그램 관계[Au] : 알고리즘과 프로그램의 관계를 이해할 수 있다.
- 문제해결과정을 말과 글로 표현[Da] : 문제 해결 과정을 말이나 글로 표현할 수 있다.
- 문제해결과정을 그림과 기호로 표현[Ab] : 문제 해결 과정을 그림이나 기호로 표현할 수 있다.
- 패턴 찾기와 표현[Ab] : 반복적으로 일어나는 일에서 패턴을 찾아 표현할 수 있다.
- 조건에 따른 해결 방법 표현[Ab] : 조건에 따라 달라지는 해결 방법을 표현할 수 있다.
- 순서도와 의사코드[Ab] : 알고리즘을 순서도와 의사코드로 표현할 수 있다.
- 알고리즘 분해와 관계[Pr] : 복잡한 알고리즘을 쪼개어 간단하게 표현하고 쪼개진 알고리즘의 관계를 파악할 수 있다.
- 다양한 순서도 표현[Ab] : 다양한 알고리즘의 구조를 순서도로 표현할 수 있다.
- 문제의 현재 상태와 목표 상태 설명[Pr] : 문제의 현재 상태와 목표 상태를 설명할 수 있다.
- 알고리즘 개선[Au] : 알고리즘의 작동 결과를 예측하고 오류수정을 통해 알고리즘을 개선 할 수 있다.
- 탐색 알고리즘 이해[Ab] : 여러 가지 탐색 알고리즘을 이해할 수 있다.
- 정렬 알고리즘 이해[Ab] : 여러 가지 정렬 알고리즘을 이해할 수 있다.
- 알고리즘 분석[Da, Ge] : 알고리즘의 장단점을 비교할 수 있다.
- 알고리즘 평가[Ge] : 알고리즘의 수행 시간이나 복잡도를 평가하고 알고리즘의 성능을 다양한 방법으로 평가할 수 있다.
- 최적 알고리즘 적용[Ge] : 최적의 알고리즘을 선택하여 유사한 문제에 적용할 수 있다.

5.2 프로그래밍

프로그래밍 영역은 일상생활의 문제를 해결하기 위해 프로그래밍 도구를 활용하여 프로그램을 작성할 수 있으며, 알고리즘의 순차, 반복, 선택 구조를 활용하여 프로그램을 작성할 수 있다. 또한, 변수, 연산자, 함수, 이벤트 등을 이용한 프로그램을 작성할 수 있는 내용으로 구성하였다.

프로그래밍 영역의 성취 기준과 관련된 컴퓨팅 사고력의 하위 요소를 설문한 결과 (Fig. 2)와 같이 분석되었다. 컴퓨팅 사고력과의 관련성이 높다고 응답한 비율이 최대값인 구성 요소를 우선적으로 선정하였다. 그러나 최대값과 차이가 5% 이내인 차순위 구성 요소가 있다면 그것을 함께 병기하였다.

Achievement criteria of programming	Pr	Da	Ab	Au	Ge
Program definition[Pr]	63.2	0.0	0.0	31.6	5.3
The relevance of program and algorithm[Au]	15.8	21.1	26.3	36.8	0.0
Program operation[Au]	31.6	15.8	15.8	36.8	0.0
Needs for programming languages[Pr]	42.1	5.3	15.8	36.8	0.0
Programming language types[Da, Au]	26.3	31.6	10.5	31.6	0.0
Simple block programming[Au]	0.0	10.5	26.3	57.9	5.3
Simple algorithm implementation[Ab]	0.0	0.0	57.9	26.3	15.8
Arithmetic operator[Au]	5.3	0.0	26.3	57.9	10.5
Compare operators and logical operators[Au]	5.3	0.0	10.5	73.7	10.5
Text programming languages[Au]	0.0	5.3	10.5	84.2	0.0
Control structures[Au]	0.0	5.3	15.8	68.4	10.5
Using variables[Au]	5.3	0.0	15.8	68.4	10.5
Program improvement[Au]	5.3	0.0	0.0	63.2	31.6
Using arrays and lists[Au]	5.3	10.5	21.1	47.4	15.8
Functions Usage[Au]	5.3	0.0	26.3	42.1	26.3
Sorting and searching[Au]	5.3	5.3	21.1	57.9	10.5
Writing a development plan[Da]	26.3	31.6	15.8	26.3	0.0
Cooperative program development[Au, Ge]	10.5	10.5	15.8	31.6	31.6
Program performance evaluation[Ge]	0.0	15.8	5.3	31.6	47.4
Understand and use the debugging of programs[Au, Ge]	5.3	0.0	15.8	42.1	36.8
Configuring the Integrated Development Environment[Au, Ge]	10.5	5.3	5.3	42.1	36.8

(Fig. 2) Programming and computing thinking

- 프로그램 의미[Pr] : 프로그램의 의미를 이해할 수 있다.
- 프로그램과 알고리즘 관계[Au] : 프로그램과 알고리즘의 관계를 이해할 수 있다.
- 프로그램 작동[Au] : 컴퓨터는 프로그램에 의해 작동됨을 이해할 수 있다.
- 프로그래밍 언어 필요성[Pr] : 프로그래밍 언어의 필요성을 이해할 수 있다.
- 프로그래밍 언어 종류[Da, Au] : 프로그래밍 언어의 종류와 장점을 이해할 수 있다.
- 간단한 블록프로그래밍[Au] : 블록 프로그래밍 언어로 간단한 프로그램을 작성할 수 있다.
- 간단한 알고리즘 구현[Ab] : 간단한 알고리즘을 프로그램으로 만들 수 있다.
- 산술 연산자[Au] : 산술 연산자를 사용하여 프로그램을 작성할 수 있다.
- 비교 연사자와 논리 연산자[Au] : 비교 연산자, 논리연산자를 사용하여 프로그램을 작성할 수 있다.
- 텍스트 프로그래밍 언어[Au] :텍스트 프로그래밍 언어로 프로그램을 작성할 수 있다.
- 제어 구조[Au] : 다양한 형태의 제어구조를 프로그램으로 작성할 수 있다.
- 변수 활용[Au] :변수를 사용하여 프로그램을 작성하고 변수범위를 이해하고 적절하게 사용할 수 있다.
- 프로그램 개선[Au] : 프로그램의 오류를 찾아 수정할 수 있다.
- 배열과 리스트 활용[Au] : 배열, 리스트을 장점을 이해하고 프로그램을 작성할 수 있다.
- 함수 활용[Au] : 함수와 라이브러리를 활용하여 프로그램을 작성할 수 있다. 함
- 정렬과 탐색[Au] : 정렬, 탐색을 위한 다양한 프로그램을 작성할 수 있다.
- 개발 계획서 작성[Da] : 문서 작성의 필요성을 알고, 적절하게 작성할 수 있다.
- 협력적 프로그램 개발[Au, Ge] :친구들과 협력하여 프로그램을 설계하고 개발할 수 있다.
- 프로그램 성능 평가[Ge] : 프로그램의 성능을 평가할 수 있다.
- 디버깅 이해와 활용[Au, Ge] : 다양한 디버깅 방법을 사용할 수 있다.
- 프로그램 개발 환경 구성[Au, Ge] : 프로그램 개발 환경을 구성할 수 있다.

5.3 로봇과 컴퓨팅

로봇과 컴퓨팅 영역은 프로그래밍이 가능한 로봇과 센서장치를 다루며 로봇 설계와 기능 및 알고리즘 구상, 로봇 또는 센서장치 부품에 대한 이해와 조립, 프로토타입 제작과 프로그래밍, 테스트와 오류 수정, 작품 발표와 공유 등으로 구성하였다.

Achievement criteria of robotics and computing	Pr	Da	Ab	Au	Ge
Definition of robot[Pr]	84.2	15.8	0.0	0.0	0.0
Types and components of robots[Da]	42.1	57.9	0.0	0.0	0.0
How robots work[Pr]	42.1	26.3	31.6	0.0	0.0
Robotics and safe use[Da]	26.3	42.1	15.8	15.8	0.0
Understanding robot behavior[Da]	21.1	36.8	21.1	15.8	5.3
Making simple motion robots[Ab, Au]	0.0	15.8	36.8	36.8	10.5
Description of simple motion robots[Ab]	5.3	5.3	52.6	21.1	15.8
Understanding rotating behavior of robots[Au]	21.1	26.3	21.1	31.6	0.0
Making various robot-driven works[Au]	0.0	10.5	5.3	68.4	15.8
Description of robot-driven works [Ge]	5.3	26.3	5.3	5.3	57.9
Making simple sensor robots[Au]	0.0	0.0	10.5	78.9	10.5
Description of simple sensor robots [Ge]	5.3	15.8	10.5	15.8	52.6
Making various sensor robot works[Au]	0.0	5.3	15.8	57.9	21.1
Description of various sensor robot works[Ge]	5.3	5.3	26.3	5.3	57.9
Rule design and robot production[Au]	0.0	0.0	21.1	73.7	5.3
Applying various rules[Ge]	0.0	0.0	15.8	26.3	57.9
Designing Robots in Daily Life[Ab, Ge]	15.8	0.0	36.8	10.5	36.8
Robot production in daily life[Ge]	5.3	0.0	10.5	36.8	47.4
Sharing and expressing robot works[Ge]	0.0	5.3	10.5	0.0	84.2

(Fig. 3) Robot & Commputing and computing thinking

로봇과 컴퓨팅 영역의 성취 기준과 관련된 컴퓨팅 사고력의 하위 요소를 설문한 결과 (Fig. 3)과 같이 분석되었다. 컴퓨팅 사고력과의 관련성이 높다고 응답한 비율이 최대값인 구성 요소를 우선적으로 선정하였다. 그러나 최대값과 차이가 5% 이내인 차순위 구성 요소가 있다면 그것을 함께 병기하였다.

로봇의 정의[Pr] : 로봇에 대해 설명할 수 있다.
- 로봇 종류와 구성[Da] :로봇의 종류와 부품, 용도에 대해 이해할 수 있다.
- 로봇 작동 원리[Pr] : 로봇이 작동되는 원리에 대해 알 수 있다.
- 로봇 법칙과 안전한 사용[Da] : 로봇의 법칙과 안전하게 사용하는 방법에 대해 이해할 수 있다.
- 로봇 동작 이해[Da] : 로봇 동작을 위해 필요한 절차를 알 수 있다.
- 간단한 동작 로봇 제작[Ab, Au] : 제시되는 순서에 따른 간단한 로봇동작을 만들 수 있다.
- 간단한 동작 로봇 설명[Ab] : 간단하게 동작하는 로봇을 설명할 수 있다.
- 회전 동작 이해[Au] : 간단한 로봇 회전 동작을 위해 필요한 부품들과 절차에 대해 이해할 수 있다.
- 다양한 로봇 구동 작품 제작[Au] : 다양한 로봇구동 작품을 만들 수 있다.
- 로봇 구동 작품 설명[Ge] : 만든 작품의 기능을 친구들에게 설명할 수 있다.
- 간단한 센서 로봇 작품 제작[Au] : 생활 속에 사용되는 센서와 로봇에 대해 설명할 수 있고 센서가 결합된 로봇 작품을 만들 수 있다.
- 간단한 센서 로봇 작품 설명[Ge] : 센서가 결합된 작품 제작 목적과 용도에 대해 설명할 수 있다.
- 다양한 센서 로봇 작품 제작[Au] : 다양한 센서들이 결합된 로봇 또는 장치를 구상과 기능을 설명하고 다양한 센서들이 결합된 로봇 작품을 만들 수 있다.
- 다양한 센서 로봇 작품 설명[Ge] : 다양한 센서들을 결합하여 만든 작품을 친구들에게 설명할 수 있다.
- 규칙 설계와 로봇 제작[Au] : 제시된 규칙을 준수하여 동작하는 작품을 설계하고 제시된 규칙을 준수하여 동작하는 작품을 만들 수 있다.
- 다양한 규칙 적용 활동[Ge] : 친구들과 규칙을 정하여 만든 작품으로 다양한 활동을 할 수 있다.
- 생활 속 로봇 작품 설계[Ab, Ge] : 생활 속 문제를 해결할 수 있는 창의적 작품을 설계할 수 있다.
- 생활 속 로봇 작품 제작[Ge] : 생활 속 문제를 해결할 수 있는 창의적인 작품을 만들 수 있다.
- 로봇 작품 공유와 표현[Ge] : 만든 작품을 친구들과 공유하고 표현할 수 있다.

6. 결론 및 제언

소프트웨어 교육의 목적은 컴퓨터과학의 개념과 원리를 이해하고, 그것을 기반으로 일상생활의 문제를 해결할 수 있는 CT를 신장시키는 것이라고 하면서도 소프트웨어 교육과 관련된 교육과정이나 운영지침에 명시된 CT의 개념은 서로 다르고, 그것을 구성하는 하위 요소도 달라, 학교 현장에서 CT를 기반으로 한 소프트웨어 교육을 정착시키는 데 어려움을 겪고 있다. 따라서 본 연구에서는 전문가 델파이 조사 결과를 토대로 CT에 대한 용어를 '컴퓨팅 사고력'으로 통일하였고, CT의 구성 요소를 문제 정의, 자료 분석, 추상화, 자동화, 일반화 등 5가지로 구분하였다. 또한, 한국정보교육학회에서 개발한 정보과 교육과정 표준 모델에서 컴퓨팅 사고력과 관련성이 높은 소프트웨어 영역을 선정하여, 하위 영역별 성취 기준과 관련성이 높은 컴퓨팅 사고력의 하위 구성 요소를 제시하였다.

따라서 본 연구에서 제시한 알고리즘, 프로그래밍, 로봇과 컴퓨팅 영역에 있는 성취 기준별 컴퓨팅 사고력의 구성 요소를 정확히 이해하고 가르친다면, 해당 내용에 포함된 컴퓨터과학의 핵심 개념 뿐만 아니라 컴퓨팅 사고력이라는 가치와 태도도 함께 기를 수 있을 것이다. 또한, 컴퓨팅 사고력은 특정 교과, 특정 영역, 특정 단원에만 한정되어 길러질 수 없으므로, 본 연구에서 제시한 예시안을 확장하여 소프트웨어 영역 이외 컴퓨팅시스템과 정보 생활 영역에도 컴퓨팅 사고력의 하위 구성 요소를 관련지어 정보과 교육과정 표준 모델의 모든 영역에서 컴퓨팅 사고력을 신장시킬 수 있도록 해야 할 것이다.

참고문헌

[1] Barefoot Computing(2016). Computational Thinking, 1. http://www.elearning.edu.my/ASKKSSM/Bahan/What%20is%20computational%20thinking.pd.

[2] Benjamin C, & Tim P.(2016). Computational thinking illustratred. http://www.ctillustrated.com.

[3] Brigitte P.(2016). School of Computer Science McGIl University, Computational Thinking: What is the science in computer science?, 3pg. http://www.cs.mcgill.ca/~bpientka/comp-thinking.pdf.

[4] CAS(2014). Developing computational thinking in the classroom: a framework. Computing At School. Developing computational thinking in the classroom.

[5] Chul Kim, Namje Park, Younghoon Sung, Soobum Shin, Youngsik Jeong(2016). Development of the Informatics Curriculums Standard Model. The Korean Association of Information Education.

[6] Code.org (2016). Retrieved from http://www.code.org.

[7] CSTA & ISTE(2011). Computational Thinking : teacher resources second edition. Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE).

[8] Denning, Peter J.(2010) The Great Principles of Computing. The Scientific Research Society. Sep-Oct, 371.

[9] Hyunchul Kim(2014). The world represented by data: the beginning of computational thinking. Seoul: Dep. of Publication in Korea University.

[10] Injoong Ju, Dongyeol Park, Misug Jin(2010). TheStudy of Core Competency's Domains and Levels. Korea Research Institute for Vocational Education & Training.

[11] Jeannette M. Wing(2011). Computational Thinking. OurCSWorkshop Carnegie Mellon University.Jeannette M. Wing(2014). Computational Thinking Benefits Society Social Issues in Computing NewYork: Academic Press. http://socialissues.cs.toronto.edu/index.html%3Fp=279.html.

[12] Ministry of Education(2015). 2015 Management Guideline of Software Education.

[13] Ministry of Education(2015). Ministry of Education(2015). Practical arts(Technology & Home Economics)/Information Curriculum.

[14] Ministry of Science, ICT and Future Planning(2011). Practical arts(Technology & Home Economics)/Information Curriculum.

[15] Quick Start Computing(2016). Computational thinking, 1. http://primary.quickstartcomputing.org/resources/pdf/comp_thinking.pdf.

[16] Seungki Shin, Youngkwon Bae(2014). Analysis and Implication about Elementary Computer Education in India. *Journal of the Korean Association of Information Education, 18*(4), 585-594.

[17] Tae-Hoon Kim, Jong-Hoon Kim, Byeong-Su Kim(2012). The Effects of Computational Thinking of Algorithm Learning using Logo for Primary Pre-service Teachers. *Journal of the Korean Association of Information Education, 16*(4), 463-474.

[18] Youngae Kim, Kapsu Kim, Jaehyun Kim, Hansung Kim, Jaemyoung Yang, Seongjin Lee, Jinmyoung Jeong, Hyunjong Choi, Kyounghwa Chae(2015). Developmnet of the Software education guideeline. Korean Education and Research Information Service.

[19] Youngsik Jeong, Jeongsu Yu, Jinsuk Lim, Youkyung Son(2015). Theory of Software Education. Cmass.

저자소개

정 영 식

1996 춘천교육대학교 수학교육학과(교육학학사)

2001 한국교원대학교 컴퓨터교육과(교육학석사)

2004 한국교원대학교 컴퓨터교육과(교육학박사)

2004~2011 한국교육개발원 연구위원

2004~현재 전주교육대학교 컴퓨터교육과 교수

관심분야: 컴퓨터교육, 프로그래밍, 이러닝

E-Mail: nurunso@jnue.kr

신 수 범

1991 인천교육대학교 (교육학학사)

1998 한국교원대학교 (교육학석사)

2002 한국교원대학교 (교육학박사)

2002~2005 KERIS 연구원

2005~현재 공주교육대학교 컴퓨터교육과 교수

관심분야: 컴퓨터교육

E-Mail: ssb@gjue.ac.kr

성 영 훈

2000. 진주교육대학교(학사)

2002. 진주교육대학교 교육대학원 컴퓨터교육 전공(석사)

2010. 경상대학교 대학원 컴퓨터과학(공학박사)

2011~2015. 한국교육학술정보원 연구원

2015~현재 진주교육대학교 컴퓨터교육과 조교수

관심분야 : SW교육, 컴퓨팅융합교육, 국가행정정보시스템

e-mail : yhsung@cue.ac.kr