https://doi.org/10.5392/JKCA.2018.18.12.238

소프트웨어 교육이 중학생의 컴퓨팅 사고력에 미치는 효과

The Effect of Software Education on Middle School Students' Computational Thinking

이정민, 고은지
이화여자대학교 교육공학과

Jeongmin Lee(jeongmin@ewha.ac.kr), Eunji Ko(kej1987@nate.com)

요약

2015 개정 교육과정은 창의·융합형 인재 양성을 목표로 정보교과에 SW교육을 포함시켜 교육과정을 운영하고 있다. 본 연구는 개정된 SW교육과정에서 추구하는 역량을 분석하여 컴퓨팅 사고력을 주요 역량으로 규정하였으며, 2018학년도 1학기에 정보교과에서 SW교육을 받은 A중학교 1학년을 연구대상으로 선정하였다. SW교육 전과 후 컴퓨팅사고력에 대한 설문조사를 실시하여 수집된 95명의 자료 중 83개의 자료가 분석에 사용되었으며, 대응표본 t-test를 통해 유의성을 확인하였다. 또한, SW교육을 통해 학습한 컴퓨팅의 개념, 수행, 관점을 산출물 기반 인터뷰를 통해 분석하였다. 분석 결과, 컴퓨팅 사고력의 하위 요인 중 비판적 사고력, 창의성, 알고리즘적 사고, 문제해결력이 유의하게 상승하였으며, SW교육 수업을 통해 새로운 개념, 수행, 관점을 습득했음을 확인하였다. 이러한 결과를 통해 본 연구는 2018년 처음 도입된 중학교 정보교과에서 소프트웨어 교육 설계 및 실행에 대한 시사점을 제공하는데 연구의 의의가 있다.

■ 중심어 : | 정보교과 | 소프트웨어 교육 | 컴퓨팅 사고력 |

Abstract

The 2015 revised curriculum includes 'informatics' course including the process of building software aiming at cultivating creative and convergent ability. This study analyzes the competencies pursued in the revised curriculum and defines computational thinking as the main competency. The subjects of the study were the first grade of a middle school in the first semester of the 2018 school year. Of the 95 collected data, 83 data were used for analysis and the significance was confirmed by the paired t-test. Also, computational concept, computational practice and computational perspectives were confirmed through artifact-based interviews. As a result of statistical analysis, critical thinking, creativity, algorithmic thinking, and problem-solving significantly increased among sub-variables of computational thinking. Statistical results and interview results were analyzed to provide implications for design and implementation of software education in 'informatics' course.

■ keyword : | Informatics | SW Education | Computational Thinking |

접수일자 : 2018년 09월 18일
수정일자 : 2018년 11월 03일
심사완료일 : 2018년 11월 03일
교신저자 : 이정민, e-mail : jeongmin@ewha.ac.kr

I. 서 론

프로그래밍이 실제적인 문제해결을 위해 문제를 정의하고 해결하는 도구로 활용되면서, 과거 한정된 전문가의 일이라는 인식에서 탈피하여 누구나 배워야 할 필수역량으로 자리 잡게 되었다. 이러한 시대적 요구에 따라 교육부는 2015 개정 교육과정을 창의·융합형 인재 양성을 목표로 하여 우리나라의 초·중등 교과과정을 역량 중심의 교과과정으로 개편하였고, 다양한 교과 내용 및 주제를 통합하여 통합형 교육과정으로 운영하고 있다[1]. 특히, 2015 개정 교육과정에서 신설된 정보(Informatics)교과는 국가 및 사회적 요구인 소프트웨어 교육 및 안전교육 강화에 대한 요구를 적극 반영하여, 소프트웨어 개발을 위한 컴퓨터 과학의 기본 개념과 원리 이해, 컴퓨팅 프로젝트 수행을 위한 코딩 기술의 습득을 포함하고, 창의적인 아이디어를 실제 프로그램이나 소프트웨어, 피지컬 컴퓨팅 장치 등을 구현하는 과정을 포함시켰다[2].

또한, 2015 개정 교육과정의 정보 교과는 실세계 문제해결을 위한 기술적·융합적 소양을 함양하는 것을 목표로 하며, 특히 정보교과 영역 중 '문제 해결과 프로그래밍'영역과 '컴퓨팅 시스템'영역의 학습을 통해 컴퓨팅사고력(computational thinking), 협력적 문제해결력의 향상을 꾀할 수 있다고 알려져 있다[1]. 컴퓨팅 사고력은 컴퓨터과학의 기본 개념과 원리, 컴퓨팅 시스템을 활용하여 실생활과 다양한 학문 분야의 문제를 이해하고 창의적으로 해법을 구현하여 적용할 수 있는 능력으로 정의되며[3], 다수 연구에서 프로그래밍 학습을 통해 추상적인 개념을 이해하고 해결방안을 찾는 과정을 거치면서 컴퓨팅 사고력 향상을 이끌 수 있다는 연구결과들이 발표되고 있다[4][5]. 또한 컴퓨팅 사고력은 복잡한 문제해결을 요구하는 현 시점에서 21세기 학습자가 갖추어야할 핵심 역량으로 논의되면서[3], 최근 컴퓨팅 사고력 향상을 위한 교수 설계 및 학습 방법에 관련된 연구가 주목받고 있다.

소프트웨어교육은 컴퓨팅 사고력을 향상시키는 효과적인 도구로 인식되고 있다[6][7]. 소프트웨어 교육은 컴퓨터의 처리과정을 이해하고 컴퓨터의 하드웨어에 대한 추상적 개념을 이해할 수 있게 함으로써 컴퓨팅 사고력을 향상시킨다고 보고되고 있고[8], 다수의 연구에서는 실증적인 효과가 규명되고 있다[9][10]. 특히, 로봇 등의 피지컬 컴퓨팅 도구를 활용한 소프트웨어 교육은 학생들이 수행한 프로그래밍 결과를 직관적으로 확인할 수 있는 장점이 있어, 소프트웨어교육을 위한 효과적인 교육적 도구로 주목받고 있다[11]. 피지컬 컴퓨팅은 현실 세계 속 데이터를 디지털 기기를 통해 소프트웨어 형태로 처리한 후, 그 결과를 LED 등의 다양한 장치로 출력하는 것을 말하는데, 프로그래밍에 서툰 학생들에게 피지컬 컴퓨팅은 컴퓨터와 프로그래밍이 동작하는 방식과 프로그래밍 원리를 쉽게 이해하게 하는 장점을 지니고 있다[12]. 따라서 로봇 등을 활용한 피지컬 컴퓨팅을 접목한 소프트웨어 교육은 학습자로 하여금 소프트웨어교육이 어렵고 지루한 것이 아니라서 놀이로 인식하게 하여 적극적으로 참여하게 하고 코딩결과를 로봇을 통해 바로 확인할 수 있게 하여, 학생들로 하여금 프로그래밍에 흥미를 느끼게 하고 학습을 지속하게 한다[13]. 따라서 연구자들은 프로그래밍에 교육용 로봇을 사용함으로써 학생들로 하여금 프로그래밍 언어와 연결하여 로봇을 구성하고 제어하도록 권장하는데[14], 이 과정에서 21세기 사회에서 요구되는 다양한 기술을 습득할 수 있다는 점에서 교육용 로봇이 강력하고 유연한 교수-학습 도구임을 알 수 있다[15]. 따라서 일부 국가에서는 교육 커리큘럼에서 프로그래밍, 팀워크, 창의력, 협업 및 컴퓨팅 사고력과 같은 학생들의 사고 기술을 개발하기 위한 혁신적 교육 도구로 로봇을 사용할 것을 권장하고 있다[16].

한편, ISTE(2015)는 인간의 문제해결 능력과 창의력, 비판적 사고력과 관련한 컴퓨팅 사고력의 의미를 재조명하고 있다[16]. 이는 학생들이 알고리즘을 사용하거나 계산을 통해 문제를 해결하는데 컴퓨팅 사고력을 활용하며, 텍스트를 분석하거나 복잡한 의사소통을 설계할 때 컴퓨팅 사고력과의 연결을 시도함을 내포한다. 이에 따라 Korkmaz과 Çakir, Özden(2017)는 컴퓨팅 사고력의 요소를 창의성, 알고리즘적 사고, 비판적 사고, 문제해결, 협력성향의 하위요소로 분류하고 이를 측정하기 위한 도구를 개발하였다[18].

본 연구에서는 2015 개정 정보 교과과정에서 추구하는 교육적 목표를 분석하여 컴퓨팅 사고력(알고리즘적 사고, 창의성, 협력, 비판적 사고력, 문제해결능력)을 소프트웨어 교육을 통해 성장 가능한 핵심 역량으로 규정하고 소프트웨어교육이 중학생의 컴퓨팅사고력을 향상시키는지 규명하고자 한다. 또한, 소프트웨어 교육을 받는 학생들이 프로그래밍 학습에 대해 어떤 인식을 가지고 있는지 확인하기 위해 산출물 기반의 심층인터뷰를 실시하였으며, 구체적인 연구문제는 다음과 같다.

연구문제 1. 소프트웨어 교육에서 중학생들의 컴퓨팅 사고력(알고리즘적 사고, 창의성, 협력, 비판적 사고력, 문제해결력)이 향상되었는가?

연구문제 2. 소프트웨어 교육을 실시함에 있어, 컴퓨팅 개념, 수행, 관점은 어떠한가?

II. 이론적 배경

1. 소프트웨어 교육

소프트웨어 교육이란 컴퓨터과학과 메타인지에 대한 이해를 바탕으로 알고리즘을 설계하고 프로그래밍하며 디버깅하는 과정을 통해 주어진 문제를 다양한 방법으로 해결할 수 있는 논리성과 창의성을 기르기 위한 교육으로 정의할 수 있다[19].

Papert(1993)는 프로그래밍 활동이 현대의 기술을 이해하고 수학과 과학에 대한 깊이 있는 접근을 위해 학습자 스스로 방법적 지식을 형성하는 구성주의적 학습에 가장 효과적인 메타인지적 활동임을 주장하였다[20]. 이는 프로그래밍 활동이 특정영역의 기술 습득을 위한 학습이 아니라 어린 시절부터 누구나 습득해야 할 인지적 능력의 습득을 위한 학습임을 강조한 것이다.

Guzdial(2004)은 학습자가 자신의 프로그램을 디버깅하면서 학습자 스스로 자신의 생각을 디버깅한다고 주장하며, 프로그래밍을 통해 고차원적 사고 능력을 얻을 수 있음을 강조하였다[21]. 뿐만 아니라 소프트웨어 교육을 통해 문제해결력 신장[22][23], 메타인지 향상[24][25], 논리적 사고력 향상[26][27]을 도모할 수 있음이 연구를 통해 밝혀지면서 소프트웨어 교육에 대한 요구는 심화되고 있는 상황이다.

이에 따라 영국, 핀란드 등의 국가는 소프트웨어 교육을 필수 교육과정으로 운영하고 있고[28], 우리나라의 경우 2014년 정부의 소프트웨어 중심사회 실현 정책을 시작으로 2015년에 소프트웨어 교육의 현장 착근을 지원하기 위한 소프트웨어교육 선도학교를 지정하여 운영하였으며, 2018년부터는 교육과정에 단계적으로 소프트웨어교육을 포함하여 교육을 실시하고 있다[1].

2015 정보과 개정 교육과정은 정보 교과의 학습 계통성을 유지하되, 창의적·융합적 사고를 형성할 수 있도록 교육과정이 개편되었다[1]. 이는 국가·사회적 요구를 반영한 것으로, 특히, 소프트웨어 교육 및 안전 교육 강화에 대한 요구를 반영하고 있고, 정보윤리 및 진로 등과 같은 범교과 학습 주제를 반영하였다.

2009 개정 교육과정에서 정보교과의 역량을 계산적 사고(computational thinking)와 정보 윤리 소양을 선정한 반면[29], 2015 개정 교육과정에서는 계산적 사고를 정보과의 핵심역량임을 가시화하기 위해 컴퓨팅 사고력으로 통일하고, 네트워크 컴퓨팅 기반 환경에서의 의사소통을 강조하기 위해 협력적 문제해결을 추가하였다[1].

2015 개정 교육과정에서 가장 큰 변화는 소프트웨어 교육 내용 중심으로 확대 개편되었다는 점인데, 문제해결과 프로그래밍 영역을 강화하고, 컴퓨팅 사고력 함양을 위한 핵심 개념으로 '추상화(abstraction)'를 도입하고 컴퓨팅 사고력 기반 문제 해결 과정을 추상화, 알고리즘, 프로그래밍으로 단계별로 제시하였다[1].

본 연구는 2015 정보과 개정 교육과정에서 문제 해결과 프로그래밍 영역을 중심으로 핵심 역량을 컴퓨팅 사고력으로 선정하였다.

2. 컴퓨팅 사고력

Wing(2006)은 컴퓨팅 사고력을 문제를 해결하고 시스템을 설계하며 컴퓨터 과학의 기본개념을 바탕으로 인간의 행동을 이해하는 것이라고 정의하면서, 컴퓨팅 사고력을 추상화와 자동화를 통한 문제해결로 설명한

다[3]. 추상화는 문제를 해결하기 위해 데이터를 분해하고 추출하여 복잡한 문제를 단순화하는 프로세스이며, 자동화는 추상화된 모델을 컴퓨터가 읽을 수 있는 프로그래밍 언어로 알고리즘화 하는 것이다[30]. 컴퓨팅 사고력에 대한 정의는 학자와 연구 기관마다 다르게 정의되고, 그 개념 또한 다르게 분류되고 있다. Wing(2006)이 추상화(abstraction)와 자동화(automation)를 컴퓨팅 사고력의 주요 개념으로 분류한 반면[3], CSTA(2011)에서는 데이터 수집(data collection), 데이터 분석(data analysis), 데이터 표현(data representation), 문제 분할(problem decomposition), 추상화(abstraction), 알고리즘과 절차(algorithms & procedures), 자동화(automation), 시뮬레이션(simulation), 병렬화(parallelization)의 과정으로 개념화 하였다[31]. 또한, 우리나라의 교육부는 CSTA(2011)가 정의한 9개 컴퓨팅 사고력 개념을 적용하여 추상화, 자동화, 창의·융합 능력을 컴퓨팅 사고력의 하위역량으로 규정하였다[1].

반면, 컴퓨팅 사고력을 측정하기 위해 몇몇의 도구가 개발되어 활용되고 있다. Brennan과 Resnick(2012)은 스크래치를 활용한 상호작용적 산출물을 제작하는 과정에서 컴퓨팅 사고력을 개념(concepts), 실행(practices), 관점(perspectives)으로 재 정의했으며[표 1], 온라인 포트폴리오 형식으로 제출된 프로젝트 결과물을 분석도구를 활용하여 평가하는 형성평가 방법과, 프로젝트 산출물 기반의 심층인터뷰, 디자인 시나리오 기법을 활용한 3가지 방법을 제안하였다[7]. 특히, 산출물 기반의 심층인터뷰는 포트폴리오 분석이나 시나리오 분석 보다 실천적인 부분에서 더 정확한 분석이 가능하다는 장점을 가지고 있다[7]. 최형신(2014)은 Brennan과 Resnick(2012)의 컴퓨팅 사고력 평가 프레임워크의 3가지 차원인 CT 개념, CT 활동, CT 관점에 기반하여 로봇 활용 프로그래밍 학습을 평가하기 위해 14문항의 설문을 개발하였다[7]. Korkmaz 등(2017)은 컴퓨팅 사고력의 수준을 측정할 수 있는 문항을 개발하기 위해 컴퓨팅 사고력의 하위 요인을 창의성, 알고리즘적 사고, 협력, 비판적 사고력, 문제해결로 나누고 총 29문항의 문항을 개발하여 활용하였다[18]. 이는 컴퓨팅 사고력을 인간의 문제 해결에 대한 기본 기술로 정의하여 문제를 해결하기 전 문제를 파악하기 위해 필요한 요소를 강조한 것으로[33], 인간이 문제를 해결하는 동안 창의성과 비판적 사고력을 발휘한다는 이점을 컴퓨터의 프로세스에 추가한 것이다[17]. 이는 컴퓨팅 사고력이 사고 기술의 종합이며[17], 컴퓨팅 사고가 곧 문제해결 과정임[34]을 강조한 결과이다.

본 연구에서는 컴퓨팅 사고력을 학생들이 문제를 형성하고 해결하는데 있어 발휘하는 사고 기술로 정의하고 중학생이라는 학습자의 수준을 고려하여 Korkmaz 등(2017)의 도구를 활용하여 컴퓨팅 사고력의 변화를 살펴보고자 하였다. 더불어, 프로그래밍 과정과 학습에 대한 인식을 구체적으로 확인하기 위해 Brennan과 Resnick(2012)의 컴퓨팅 사고력 개념에 입각한 산출물 기반 인터뷰를 실시하였다.

표 1. Brennan & Resnick(2012)의 컴퓨팅사고력의 차원 분류

차원	구성요소
concept	sequences, loops, parallelism, events, conditionals, operators, data
practice	incremental/iterative, testing/debugging, reusing/remixing, abstracting/modularizing
perspective	expressing, connection, questioning

3. 컴퓨팅 사고력과 소프트웨어 교육

K-12 교육과정에서 컴퓨팅 사고력의 향상은 다음과 같은 잠재력을 갖는다. 첫 번째, 컴퓨팅 사고력을 향상시킴으로써, 학생들로 하여금 컴퓨팅 과정을 이해하게 하고, 정보를 추상화하고 표현할 수 있는 능력을 기르게 할 수 있다[35]. 두 번째, 학생들이 복잡한 문제를 다루는데 자신감을 주고, 공동의 목적을 달성하거나 해결책 고안을 위해 타인과 의사소통하는 능력을 길러준다[36].

소프트웨어 교육을 통한 컴퓨팅 사고력의 향상은 몇 가지 실증연구를 통해 살펴볼 수 있다. 이철현(2017)은 컴퓨팅 사고력 기반 실생활 문제해결학습 모형을 수립하여 초등학생에게 프로그램을 적용한 결과, 프로그래밍 학습이 학습자의 컴퓨팅 사고력 향상에 긍정적인 효과가 있음을 확인하였다[10]. 또한, 서영호 등(2016)은 두 사람이 하나의 컴퓨터를 사용하여 협력적 분업 형태

로 프로그래밍을 하는 동료 프로그래밍 교육방법을 실험집단에 적용한 결과, 강의식 학습을 진행한 비교집단에 비해 컴퓨팅 사고력 신장에 있어 유의한 상승이 있음을 확인하였다[9].

언어 기반의 프로그래밍을 통해 컴퓨팅 사고력이 향상되기도 하지만 피지컬 컴퓨팅 도구가 활용되었을 때 더 효과적일 수 있다. 김재휘와 김동호(2016)는 블록형 프로그래밍 언어를 사용한 집단과 센서보드를 활용한 피지컬 컴퓨팅 교육과정을 적용한 집단을 비교한 결과, 피지컬 컴퓨팅 교육과정을 진행한 집단의 컴퓨팅 사고력 요소 중 자료표현, 수행 및 검증, 일반화에서 유의하게 더 큰 향상을 보였다고 보고하였다[37].

로봇을 활용한 프로그래밍 학습 또한 컴퓨팅 사고력 향상에 효과적임이 다수의 연구를 통해 확인되고 있다. Atmatizidou와 Demetriadis(2016)은 중학생과 고등학생을 대상으로 레고 마인드스톰 교육용 로봇 키트를 사용하여 교육을 진행하였는데, 중학생과 고등학생 집단 모두 컴퓨팅 사고력의 유의한 향상을 나타냈다[38]. 또한, Lee 등(2011)은 로봇 프로젝트에서 학생 프로그래머가 로봇이 환경과 상호작용하는 것을 프로그래밍하며 추상화와 자동화 같은 컴퓨팅 사고 능력을 향상시킬 수 있었다고 보고 하였다[39]. 이정민 등(2018)은 교육용 로봇인 알버트(Albert)를 활용한 프로그래밍 수업을 통한 초등학생의 컴퓨팅 사고력의 향상을 입증함으로써 로봇을 활용한 프로그래밍 학습이 컴퓨팅 사고력 향상에 효과적인 교수-학습 환경임을 주장하였다[40].

Korkmaz 등(2017)이 제시한 컴퓨팅 사고력 하위요인인 창의성, 비판적 사고, 알고리즘적 사고, 협력, 문제해결에 소프트웨어 교육이 결정적임이 선행연구를 통해 드러나고 있다. Khanlari(2013)은 로봇을 활용한 소프트웨어 교육 과정에 이전에 존재하지 않았던 것을 창조해내거나 모델을 발전시키는 과정에서 협력, 비판적 사고, 창의성과 같은 21세기 학습자 역량을 향상시킬 수 있다고 주장한다[41]. 전성균 등(2011)은 프로그래밍을 학습하는 과정에서 문제를 인식하여 문제해결을 위한 다양한 방법을 찾아보고, 오류 발생 시 오류 수정을 위해 다양한 사고를 시도하기 때문에 소프트웨어 교육이 창의성을 향상시키는 강력한 도구라고 주장하였으며[42], 박경재와 이수정(2010)의 연구에서 두리틀과 로봇 교육의 효과성 확인을 위해 실험연구를 실시한 결과, 일반 수업반에 비해 두리틀, 로봇 교육반의 창의성이 크게 향상되었음을 확인하였다[43]. 문제해결력 향상에 대해 배영권과 남재원(2010)은 로봇 프로그래밍 교육이 문제해결력 신장에 긍정적 영향을 미치는 것을 확인하였으며[22], 송정범 등(2008)은 창의적 문제해결 수업모형을 토대로 스크래치를 활용하여 소프트웨어 수업을 진행할 때 학습자의 문제해결력이 향상되었음을 확인하였다[23]. 이 외에도 프로그래밍 학습 자체가 알고리즘적 프로세스를 실제로 체험할 수 있는 도구이며[44], 교육용 프로그래밍 언어인 스크래치가 어린아이들에게 알고리즘을 가르치기 위해 개발되었음을 선행연구를 통해 확인할 수 있다[45].

위와 같은 선행연구에 따라 본 연구에서는 다양한 소프트웨어 교육을 통해 컴퓨팅 사고력을 향상시킬 수 있을 것이라 가정하며, 학습 전 후를 비교하여 그 효과성을 확인하고자 한다.

4. 수업설계

본 연구를 위해 정보 수업에서 소프트웨어 교육은 다음과 같이 설계되었다[표 2].

표 2. 주차별 학습 주제 및 학습내용

주	시수	학습 주제	학습내용	비고
1	1	스크래치와 하드웨어	코드이노 설정하기 스크래치와 연결하기	코드이노 스크래치
2	2	전자기타	저항 설정하기 기타 코딩하기	메이키 메이키
3	2	목소리 그래프	소리센서 코딩하기	코드이노
4	2	컴퓨터의 구성과 동작	컴퓨팅 시스템을 구성하는 하드웨어와 소프트웨어의 역할을 이해하고 유기적인 상호관계를 분석한다.	엔트리봇 폭탄넘기기
		헬리콥터 피하기	엔트리봇과 폭탄넘기기 게임학습을 통하여 컴퓨팅 시스템의 구성과 동작원리를 협력학습으로 익히기 3축센서 코딩하기	
5	2	빛으로 켜지는 선풍기	빛 센서 코딩하기	코드이노
6	2	신기록 세워라	버튼 센서 코딩하기	코드이노
7	2	카레이싱		

	2	새로운 정보 기술의 윤리적 활용	3D 모델링 틴커캐드를 활용하여 원하는 모델링을 만들고, 출력과정을 이해할 수 있다.	틴커캐드
8	1	목각인형극	무대 코딩하기	코드이노 메이키 메이키
9	1	충격감지기		
10	1	소리조절 스피커	스피커 코딩하기	코드이노 메이키 메이키
11	2	픽맨		
	2	아날로그와 디지털의 구분 및 특징	일상생활에서 사용하는 물건을 아날로그 장치와 디지털 장치로 구분할 수 있다. 현실 세계의 다양한 자료와 정보를 디지털 정보로 변환하는 방법을 설명할 수 있다.	햄스터 로봇
12	1	햄스터로봇	햄스터로봇을 활용한 기본 코딩 학습하기 조암올림픽경기를 기획하여 나만의 올림픽경기를 기획하고 규칙 만들고 구현할 수 있다.	햄스터 로봇
13	2	햄스터로봇		
14	2	햄스터로봇	햄스터로봇 힘겨루기 대회	햄스터 로봇
15	2	햄스터로봇		
16	1	알고리즘	최단거리 알고리즘 생각해보기	리코셰 로봇

2018학년도 1학기 정보 수업에서는 스크래치, 엔트리 등의 프로그래밍 언어와 코드이노, 메이키 메이키 등의 센서보드, 3D프린터, 그리고 엔트리봇, 햄스터 로봇, 리코셰로봇 등의 로봇을 활용하여 수업이 구성되었으며, 소프트웨어 교육은 정보교과 총 17주(34시간) 중 30시간 동안 운영되었다.

Ⅲ. 연구 방법

1. 연구 대상

본 연구를 위해 2018년도 1학기에 경기도 소재의 A 중학교 1학년 4개 학급의 학생 95명을 대상으로 정보교과 커리큘럼에 소프트웨어 교육을 포함하여 수업을 진행하였다. A 중학교는 SW 교육 선도학교이며, 정규교과와 더불어 소프트웨어 동아리를 운영하여 SW 역량 강화에 힘쓰고 있다. 또한, 모든 학생은 남학생이다.

본 연구를 위해 수집된 양적 데이터 중 83개의 데이터가 분석에 사용되었으며, 효과성 검증을 위해 단일집단 사전-사후 검사로 설계되었다. 검사 과정에서 발생할 수 있는 시험효과(testing effect)를 방지하기 위해 사후 검사는 사전 검사 후 17주의 충분한 간격을 두고 실시하였으며, 사후 검사의 문항 순서를 무작위로 변경하여 시행하였다.

또한, 소프트웨어 교육 경험의 질적 자료 수집을 위해 인터뷰를 실시하였으며, 총 12명의 학생이 인터뷰에 자발적으로 참여하였다. 인터뷰 시간은 학생당 10분 정도가 소요되었다.

2. 측정 도구

본 연구에서는 컴퓨팅 사고력 측정을 위해 Korkmaz 등(2017)에 의해 개발된 컴퓨팅 사고력 측정도구를 사용하였다. 이 도구는 창의성, 알고리즘적 사고, 협력, 비판적 사고, 문제해결 등의 5개의 하위 요인으로 구성되어 있다[18]. 이 도구는 연구의 맥락에 맞게 수정되었으며, 교사 1인과 교육공학 교수에 의해 내용 타당도를 검증받았다. 창의성 8문항, 알고리즘적 사고 6문항, 협력 4문항, 비판적 사고 5문항, 문제해결 6문항, 총 29문항으로 구성되어 있으며, 모든 문항은 5점 Likert 척도로 구성하였다. 본 연구에서 검증된 신뢰도 Cronbach의 α 값은 .94로 나타났다.

또한, 학생들이 수행한 소프트웨어교육 프로젝트 내용을 살펴보면서 컴퓨팅 사고력에 대한 3가지 프레임워크(개념, 수행, 관점)에 관련된 문항으로 산출물 기반 인터뷰를 실시하였다. 인터뷰는 프로그래밍에 대한 생각과 수행과정에 대한 질문, 타인과의 협력여부, 프로젝트에 대한 아이디어의 근원과 창작활동에 대한 질문, 선호하는 프로그래밍 특징과 기술 등을 포함한다.

표 3. 측정도구 구성

하위 영역	문항 예시	문항 수	Cronbach's α
창의성	나는 새로운 상황에서 발생하는 문제를 잘 해결할 수 있다.	8	.80
알고리즘적 사고	나는 문제에 대한 해결책을 찾았을 때, 그 해결책이 맞는지 맞지 않는지 즉시 판단할 수 있다.	6	.87
협력	나는 친구들과 문제를 해결하는 것을 좋아한다.	4	.88
비판적 사고	나는 주어진 방법의 장단점을 체계적으로 비교하면서 결정을 내릴 수 있다.	5	.81
문제해결	나는 내가 세운 계획을 짜임새 있게 순서대로 적용하는 방법을 알고 있다.	6	.85

표 4. 인터뷰 문항 구성

영역	질문
산출물 관련 질문	정보시간에 배운 프로그래밍 활동 중 가장 기억에 남는 활동이 무엇인가요?
	본인이 만든 프로젝트의 주제에 대해 말해주세요.
	이 프로젝트의 아이디어는 어떻게 얻게 되었나요?
컴퓨팅 개념	조건에 따라 다르게 수행하는 부분을 어떻게 프로그래밍 했나요?
	반복되는 부분은 어떻게 프로그래밍 했나요?
	변수를 사용하면 좋은 점이 무엇인가요?
	본인의 프로젝트는 어떤 방법으로 프로그래밍 했나요?
컴퓨팅 수행	프로그래밍을 할 때, 중간에 테스트를 해보나요, 아니면 다 만들고 테스트를 해보나요?
	프로그램에 오류가 났을 때 수정한 적이 있나요? 있다면, 어떻게 해결했나요?
	다른 친구들의 도움을 받거나 친구들의 프로그램을 활용해 본 경험이 있나요?
	본인의 프로젝트 안에 구성된 각각의 기능을 구분할 수 있나요?
컴퓨팅 관점	프로그래밍을 좋아하는 이유는 무엇인가요? 어떤 부분을 가장 좋아하고, 그 이유는 무엇인가요?
	프로그래밍을 통해 나 자신을 더 창의적으로 표현할 수 있다고 생각하나요? 그 이유는 무엇인가요?
	프로그래밍을 할 때, 친구들과 함께하는 것이 더 좋은 결과를 만들 수 있다고 생각하나요? 그 이유는 무엇인가요?

3. 자료분석방법

본 연구에서 수집된 양적 자료는 통계패키지 SPSS를 이용하여 분석되었으며, 다음과 같은 절차를 거쳤다. 첫째, 설문 문항의 내적 일관성을 검증하기 위해 Cronbach의 α 계수를 확인하였다. 둘째, 대응표본 t-검증을 실시하여 정보교과에서 소프트웨어 교육이 중학생의 컴퓨팅사고력에 미치는 효과를 살펴보았다. 모든 연구 결과는 유의수준 .05 수준에서 분석되었다.

또한, 질적 분석을 위해 산출물 기반 심층 인터뷰를 실시한 후 인터뷰 내용을 전사한 후, 학생들의 이해도와 인식을 범주화하기 위해 개방형 코딩을 실시하여 분석하였다.

Ⅳ. 연구 결과

1. 컴퓨팅 사고력 하위요인에 대한 대응표본 t-검정

정보 교과에서 소프트웨어 교육을 진행하기 전·후에 설문을 통한 컴퓨팅 사고력의 하위요인인 창의성, 알고리즘적 사고, 협력, 비판적 사고, 문제해결력 검사를 실시하였으며, 각 요소에 대한 대응표본 t-검정을 수행한 결과는 다음과 같다[표 5].

컴퓨팅 사고력의 하위 요인 중 창의성은 사전 3.39에서 사후 3.93으로 .54점이 상승한 것으로 나타났으며, 이는 유의수준 .05에서 유의한 차이로 나타났다(t=-3.06, p<.05). 또한, 알고리즘적 사고는 사전 3.25에서 사후 3.58로 .33점이 상승한 것으로 나타났으며, 이는 유의수준 .05에서 유의한 차이로 나타났다(t=-3.79, p<.05). 비판적 사고력은 사전 3.43에서 사후 3.67로 .24만큼 상승하였으며, 이는 유의수준 .05에서 유의한 차이로 나타났다(t=-2.89, p<.05). 또한, 문제해결력은 사전 3.53에서 사후 3.73으로 .20만큼 상승하였으며, 이는 유의수준 .05에서 유의한 차이로 나타났다(t=-3.22, p<.05).

그러나 협력의 평균은 사전 4.04에서 사후 4.04로 사전-사후 간 차이가 없는 것으로 나타났다.

표 5. 대응표본 t-검정 결과

하위 요인	사전		사후		t	p
	M	SD	M	SD		
창의성	3.69	.57	3.93	.61	-3.06*	.003
알고리즘적 사고	3.25	.77	3.58	.88	-3.79*	.000
협력	4.04	.79	4.04	.85	.047	.962
비판적 사고	3.43	.72	3.67	.78	-2.89*	.005
문제해결력	3.53	.68	3.73	.73	-2.79*	.007
Total	3.57	.60	3.79	.64	-3.22*	.002

*p <.05

2. 산출물 기반 인터뷰

학생들은 정보 수업에서 배운 프로그래밍 활동 중 가장 기억에 남는 활동으로 3D프린팅과 햄스터 로봇활용 프로젝트를 선정하였다. 3D프린팅을 선정한 이유로, “프린팅이 실제로 되는게 신기했고, 처음봐서 색달랐어요”, “실생활에 유용할 것 같고, 만들기 쉬울 것 같아서요.”, “커서 돈 벌어서 직접 집지으려고요, 제가 살고 싶은 집을 디자인했어요.” 등의 응답을 했다. 즉, 3D 프린팅을 통해 실생활에 유용한 물건(물체)을 직접 디자인하여 만들 수 있다는 점과, 3D프린팅이 기존에 배웠던 프로그래밍 활동과 차별화된 경험이라는 점을 이유로

대답하였다. 또한, 햄스터를 활용한 프로젝트를 선정한 이유로는 “제 생각을 컴퓨터에 입력했는데, 입력을 한 대로 햄스터가 똑같이 따라해주니까 신기했어요.” 등의 햄스터 자체에 대한 흥미와 ‘미로탈출’, ‘올림픽’ 등의 프로젝트가 자신에게 익숙한 주제였다는 점을 꼽았다. 학생들은 햄스터를 활용한 프로젝트로 이어달리기, 격투기, 레이스 경주, 힘겨루기 등을 수행하였는데, 이러한 프로젝트 주제를 선정한 배경에는 과제의 난이도와 친구들과의 협력 여부가 결정적인 것으로 나타났다.

컴퓨팅 ‘개념’에 대한 질문은 조건문과 반복문의 수행, 변수의 활용, 학생의 프로젝트에 대한 프로그래밍 방법으로 구성되었다. 먼저, 조건에 따라 다르게 수행되는 부분을 어떻게 프로그래밍 했는지 물어보는 질문에 대해, 특정한 키보드를 누를 때 어떻게 움직이게 할 것인지 명령을 부여하거나 어떤 조건을 마주할 때(예: 벽에 부딪혔을 때) 특정 행동(예: 90도로 회전, 양쪽 바퀴 속도를 조절하여 회전)을 하게 하는 등의 조건문(if, if~else)을 활용하였다고 대답하였다. 이 과정에서 몇몇 학생들은 “조합해서 했어요, 만약에 왼쪽으로 가야하는데 안된다면 90도 꺾는 블록을 사용하고...”, “바퀴 속도를 조절하거나, 바퀴 움직임을 정지시키거나 하는 방법으로 회전시켰어요.”, “하나하나씩 블록을 맞춰보고 틀리면 빼고 시도했어요.” 등의 응답을 하였는데, 학생들이 문제를 해결하는데 있어 자연스럽게 시행착오를 경험하며 해결점을 찾는 방법을 경험하고 있음을 확인할 수 있었다. 반복되는 부분에 대해 어떻게 프로그래밍 했는지 물어보는 질문에는, 인터뷰에 참여한 대부분의 학생이 반복하기, 무한반복하기 등을 활용해야 하는 것을 정확하게 알고 있었다. 변수를 활용할 때의 이점에 대해서는 몇몇 학생들이 “더 간단하게 코딩을 빨리 끝낼 수 있어요”라고 대답하여 코드의 간결화를 위해 변수를 사용할 수 있다는 장점을 인식하고 있는 반면, 대부분의 학생이 ‘변수’라는 개념 자체를 이해하지 못하는 것으로 나타났으며, ‘변수’의 개념을 알고 있는 학생도 왜 변수를 사용해야하는지 그 필요성에 대해서 인지하지 못하고 있었다. 이는 복잡한 프로젝트를 경험해보지 못한 결과로도 해석해 볼 수 있다. 자신의 프로젝트를 설명해 달라는 질문에 대해서는 “처음에 오르막길이 있어서 최대로 빠르게 해서 올라간 다음에, 오른쪽으로 돌아야하니까 오른쪽바퀴를 왼쪽바퀴보다 느리게 해서 돌고 왼쪽으로 돌아야하면 왼쪽바퀴를 느리게 했어요. (왼쪽바퀴를 느리게 하는 건 어떻게 했어요?) 속도 바꾸는 방법이 있는데..왼쪽바퀴를 30으로 정하면 오른쪽 바퀴를 100으로 바꿔서 했어요.” 라고 대답하는 등 대부분의 학생이 자신의 프로젝트에 대해 자신 있게 설명하였으며, 교사가 제공한 기본적인 코드를 친구들과 상의하거나 인터넷에서 방법을 찾아봄으로써 변형·심화시켜 활용하는 경우가 많음을 확인할 수 있었다.

컴퓨팅 ‘수행’에 대한 질문은 테스트, 오류 수정, 코드의 재사용, 기능의 구분 등이 포함되어 있다. 먼저, 학생들의 테스트 유형을 질문한 결과, 약 14%를 제외한 학생들이 프로그래밍 중간 중간에 테스트를 해보는 경향을 보였다. “중간 중간에요. 만약에 다 완성되고 테스트했는데 안 되면 처음부터 해야 하니까 중간 중간 점검을 하는 게 좋은 것 같아요.”, “중간 중간에 테스트해요. 마지막에 한 번에 하면 어디가 틀린지 모르는데 중간 중간에 하면 어디가 틀린지 알 수 있어요.”라고 대답한 내용을 미루어 보아, 이들은 프로그래밍 중간에 테스트를 해볼 때 오류가 발생하더라도 어디가 잘못되었는지 정확하게 파악할 수 있는 장점을 알고 있었다. 이 과정에서 오류 수정은 오류가 발생한 코드만 따로 떼어내어 수정하거나 전체적으로 다른 방법을 생각해보는 경향을 보였다. 또한, 다른 친구들의 도움을 받거나 프로그램을 활용해본 경험이 있는지에 대한 질문에, 한 명의 학생을 제외한 모든 인터뷰 참가자들이 그렇다고 대답했다. 자신이 모르는 개념에 대해 물어보거나, 다른 학생이 하는 방법이 간편해보여서 따라한 적이 있다는 응답과 다른 학생이 만든 프로그램이 재미있어보여서 함께 해본 적이 있다는 응답이 있었다. 본인의 프로젝트 안에 구성되어 있는 작은 부분(기능)을 구분할 수 있냐는 질문에는 대부분의 학생이 각 부분이 어떤 기능을 하는지 설명할 수 있다고 응답한 반면, 몇몇 학생들은 실행을 통해 확인해봐야 알 수 있다는 반응을 보였다.

컴퓨팅 ‘관점’에 대한 질문은 프로그래밍 선호도, 창의성 발현, 협력에 관련된 질문으로 이루어져 있다. 프로그래밍을 좋아하냐는 질문에 인터뷰에 참여한 모든

학생들은 프로그래밍에 대해 “제가 설정하는 대로 움직일 수 있다는 점이 흥미로워요.”, “예전에는 컴퓨터 안에서만 가능했는데, 로봇이 나오면서 더 재밌게 코딩할 수 있는 것 같아요.”, “스스로 여러 가지 생각해보면서 창의력도 길러지는 것 같아요.” 등의 긍정적인 태도를 보였다. 또한, 몇몇 학생들은 퍼즐처럼 조각조각 맞추어 여러 가지 기능을 구현할 수 있었다는 점, 쉬운 코딩부터 어려운 코딩까지 단계적으로 배울 수 있다는 점을 프로그래밍을 좋아하는 이유로 대답하였고, 몇몇의 학생은 코딩 교육을 통해 미래의 직업을 준비할 수 있다는 기대감을 보였다. 프로그래밍을 통해 자신을 더 창의적으로 표현할 수 있냐는 질문에는 대부분의 학생이 긍정적인 답변을 보였으나, 일부 학생은 “다른 공부를 할 때랑 별로 차이를 못 느끼겠어요.” 라고 대답하는 등 부정적인 응답을 내놓기도 하였다. 긍정적인 답변에는 “내 머릿속에 있는 것을 블록을 이용해서 그대로 나타낼 수 있으니까요.”, “프로그램을 만들면 보완하기 위해서 새로운 아이디어를 생각해야 하니까요.”, “남들이 시도하기 어려운 것을 나는 시도해볼 수 있으니까요.”, “만약에 하다가 마음에 안 들더라도 다른 것을 만들면 되니까요.” 등이 있었다. 프로그래밍을 할 때 협력을 선호하냐는 질문에는 긍정적인 답변과 부정적인 답변으로 의견이 나뉘었다. 긍정적인 답변에는 친구가 모르는 것을 해결해 주거나 오류를 찾아줄 수 있고, 경쟁구도로 하나의 프로그래밍을 하면 더 발전적인 결과가 나오며, 흥미가 극대화 된다는 점이 있었다. 반면, 부정적인 답변에는 친구들이랑 할 때는 집중이 되지 않는다는 점, 자신의 성향 상 무언가를 만들 때 소통하는 것을 선호하지 않는다는 점 등이 있었다.

V. 결론

본 연구는 소프트웨어 교육이 중학생의 컴퓨팅 사고력에 미치는 영향을 확인하기 위해 수행되었다. 연구결과, 소프트웨어 교육은 중학생의 컴퓨팅 사고력 중 창의성, 문제해결력, 비판적 사고능력, 알고리즘적 사고능력에 대한 인식을 유의하게 증가시켰으며, 통계적 결과와 인터뷰 결과에 따른 구체적인 논의는 다음과 같다.

첫째, 학생들의 문제해결능력과 비판적 사고능력에 대한 인식이 소프트웨어 교육을 통해 증가되었다. 이는 문제를 해결함에 있어 프로그래밍을 활용할 때 비판적 사고력, 문제해결능력과 같은 고등사고능력이 향상될 수 있음을 논의한 선행연구의 견해와 일치한다[17][21-23][46]. 또한, 본 연구를 위해 설계된 커리큘럼에 포함된 피지컬 컴퓨팅과 로봇이 문제 해결 상황에서 즉각적인 피드백을 제공하는 것과 관련지어 생각해 볼 수 있다. 인터뷰 내용과 같이 학생들은 코딩 중간 중간에 테스트를 수행하여 시각화된 상황을 살피고, 오류가 발생했을 때 또 다른 전략을 구상하여 실행해 보는 등 시행착오 과정을 겪으면서 복잡한 문제를 비판적으로 해결할 수 있는 능력을 습득하였다고 스스로 인식하게 된 것이다. 따라서 교사는 학생들의 실패에 있어 직접적으로 해결해주거나 문제해결에 직접적으로 개입하기보다는 학생 스스로 다른 전략을 활용할 수 있도록 독려하는 역할을 수행해야 한다.

둘째, 학생들은 소프트웨어 교육을 통해 자신의 창의성을 향상되었다고 인식하였다. 이는 소프트웨어 수업에서 학생들이 이전에 존재하지 않았던 새로운 것을 창조해내거나 혁신적인 방법을 사용하여 모델을 효율적으로 발전시켰다는 연구결과와[41] 프로그래밍을 통해 문제를 해결하는 과정에서 창의성을 발휘한다는 선행연구의 견해[17][43]와 일치하는 결과이다. 인터뷰 결과를 참고하면, 학생들이 소프트웨어 교육을 통해 머릿속에 있는 생각을 구체적인 조작을 통해 표현할 수 있으며, 여러 번 수정 가능하다는 프로그래밍의 장점은 학생들로 하여금 더욱 발전적인 아이디어를 생산하고 적용하게 함을 알 수 있다. 교사는 이러한 프로그래밍의 장점을 고려하여 단순하고 일회적인 과제를 제시하기보다는 자신의 아이디어를 지속적으로 구현할 수 있도록 수업 설계에 있어 충분한 시간과 기회의 제공을 고려해야 할 것이다.

셋째, 학생들은 자신의 알고리즘 사고력이 향상되었다고 인식하였으며 이는 선행연구의 견해와 일치한다[17][45]. 스크래치와 엔트리를 활용한 프로그래밍 과정에는 자신이 원하는 결과를 구현하기 위해 적절한 순서

와 절차로 블록을 쌓아가는 활동을 포함한다. 만약 학생들이 코드를 순차적으로 제시하지 않거나 적절한 블록을 사용하지 않는다면 프로그래밍 결과는 오류를 발생시키는 등의 즉각적인 반응을 하게 되는데, 이와 같은 과정에서 학생들은 오류를 수정하는 과정을 거치며 알고리즘을 학습할 수 있다. 학생들의 인터뷰 중 스크래치와 엔트리를 활용함에 있어 자신이 원하는 움직임을 구현하기 위해 조건문과 반복문을 활용하고, 적절한 수치를 찾아가는 과정에서 오류를 수정하는 경험을 했다는 내용을 참고하면 프로그래밍 학습 과정이 곧 알고리즘적 사고의 습득 과정이라고 볼 수 있다.

반면, 소프트웨어 교육이 협업 및 의사소통에 있어 효과적이라는 선행연구[36][41]의 결과와는 달리, 본 연구에서는 학생들의 협력성향의 변화가 유의하지 않았다. 이러한 결과는 학생들의 인터뷰를 통해서도 확인할 수 있는데, 일부 학생들은 소프트웨어 교육 시 개인적으로 집중하는 시간을 선호하는 성향을 가지고 있었기 때문이라고 볼 수 있다. 그러나 동료의 도움을 받아 오류를 수정할 수 있다는 점과 경쟁구도를 통해 흥미가 극대화 될 수 있다는 점을 협력적 프로그래밍의 장점으로 제시한 학생들의 의견을 미루어 보아, 장기적인 소프트웨어 교육은 협력성향의 향상에 긍정적인 영향을 미칠 것으로 짐작해 볼 수 있다.

위와 같은 연구결과를 종합하여, 다음과 같은 시사점을 제공하고자 한다.

첫째, 각 프로그램을 활용한 소프트웨어 교육이 일회성으로 끝나지 않도록, 보충 및 심화 과정을 수업에 포함시키는 것을 고려해야 한다. 지식을 전달하는 방식의 수업은 학생들이 단순한 프로그래밍 기술을 습득하는 것에 그치게 한다. 심화된 프로젝트의 진행은 다양한 아이디어 생산과 여러 가지 프로그래밍 시도를 통한 사고능력 함양에 도움이 될 것이다.

둘째, 교사는 학생들이 가진 문제를 해결하는데 가이드로서의 역할을 수행해야 한다. 학생 스스로 시행착오를 겪을 때 고차원적 사고능력이 향상되기 때문에, 학생이 스스로 다른 아이디어나 전략을 생각해보도록 교사는 그 방향을 안내하는 역할을 수행해야 할 것이다.

셋째, 정규수업 뿐만 아니라 동아리, 방과 후 수업 등과 연계하여 충분한 학습 시간을 확보해야 한다. 정보교과의 한 학기 수업 시수는 34차시로 구성되어 있으며, 이는 다양한 프로그램을 학습하기에는 다소 부족한 시간이다. 수업의 특성 상 기본적인 개념을 학습하는데 그칠 수 있기 때문에, 동아리나 방과 후 수업을 통해 수업에서 학습한 내용을 바탕으로 심화된 프로젝트를 수행할 수 있는 시간을 확보하는 것이 중요하다.

본 연구는 현재 교육부에서 제시하고 있는 소프트웨어 교육 목표를 바탕으로 개정된 교육과정의 초기 단계에서 소프트웨어 교육의 효과를 확인하여 시사점을 제공한데 연구의 의의가 있다.

본 연구의 의의와 시사점에도 불구하고 몇 가지 한계점이 있다. 첫째, 한 학교만을 대상으로 연구를 수행했다는 점, 연구를 수행한 학교가 이미 소프트웨어 교육 선도학교로 지정되어 있었다는 점, 남학생만으로 구성되어 있다는 점에서 연구의 일반화에 한계가 있다. 지역, 성별에 따라 소프트웨어교육의 차이가 있을 수 있으므로 이를 고려한 후속연구가 필요하다. 둘째, 컴퓨팅 사고력 향상의 측정을 자기보고식 설문과 주관적인 판단이 포함된 인터뷰로 측정하였기 때문에 학생 스스로 인지하고 있는 자신의 능력과 실제 사고력에는 차이가 존재할 수 있다. 현 소프트웨어 교육 운영 지침에서 다양한 평가 방법을 사용하도록 권고하고 있는 만큼, 자기보고식 설문과 함께 객관적인 사고력 측정을 위해 비버챌린지 등과 같은 과제 해결 중심의 평가 모델을 활용하거나 외부 관찰자에 의해 평가되는 과정이 포함될 필요가 있다. 셋째, 본 연구에서 설계된 교육과정에는 장기적인 협력 교육프로그램이 포함되지 않아, 일회적인 협력만으로 협력성향의 변화를 측정하기 어려웠다. 소프트웨어 교육이 협력성향에 미치는 영향을 규명하기 위해서는 보다 장기적이고 협력주제 중심의 교육과정을 구성하여 연구할 필요가 있다.

참 고 문 헌

[1] 교육부, *2015 개정 교육과정 총론*, 교육부 고시 제 2015-74호 (별책 1), 2015.

[2] 김경훈, 이은경, 김영애, 양재명, 이영준, 김현철, 김재현, 배정이, 한건우, 박소영, 박종훈, 정웅열, 이진, 김영아, 장원영, *2015 개정 교과 교육과정 시안 개발 연구 II-정보과 교육과정*, 한국교육과정평가원 연구보고 CRC 2015-25-14, 2015.

[3] J. M. Wing, "Computational Thinking," Computations of the ACM, Vol.49, No.2, pp.33-35, 2006.

[4] 황요한, 문공주, 박윤배, "소프트웨어 활용 탐구 활동을 통한 고등학생의 프로그래밍과 컴퓨팅 사고력에 대한 인식 변화와 과학 학습에 대한 태도 조사-스크래치와 피지컬 컴퓨팅 교구의 활용을 중심으로," 한국과학교육학회지, 제36권, 제2호, pp.325-335, 2016.

[5] A. OLUK and Ö. KORKMAZ, "Comparing Students' Scratch Skills with Their Computational Thinking Skills in Terms of Different Variables," International Journal of Modern Education & Computer Science, Vol.8, No.11, pp.1-7, 2016.

[6] M. Berland and U. Wilensky, "Comparing Virtual and Physical Robotics Environments for Supporting Complex Systems and Computational Thinking," Journal of Science Education and Technology, Vol.24, No.5, pp.628-647, 2015.

[7] K. Brennan and M. 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, pp.1-25, 2012.

[8] 오미자, 김미량, "컴퓨팅 사고력 향상을 위한 스크래치 프로그래밍 교육의 효과 분석," 교육정보미디어연구, 제24권, 제2호, pp.255-275, 2018.

[9] 서영호, 염미령, 김종훈, "초등학교 SW교육에서 동료 프로그래밍 교육 방법이 컴퓨팅 사고력과 창의성 신장에 미치는 효과 분석," 정보교육학회논문지, 제20권, 제3호, pp.219-234, 2016.

[10] 이철현, "컴퓨팅 사고력 기반 실생활 문제해결학습이 초등학생의 컴퓨팅 사고력에 미치는 효과," 실과교육연구, 제23권, 제4호, pp.91-107, 2017.

[11] 김혜진, 서정현, 김영식, "아두이노를 연계한 스크래치 프로그래밍 교육이 중학생의 창의적 문제해결력에 미치는 영향," 학습자중심교과교육연구, 제16권, pp.707-724, 2016.

[12] D. O'Sullivan and T. Igoe, *Physical Computing: Sensing and Controlling the Physical World with Computers,* Boston, MA, USA: Thomson, 2004.

[13] 최유현, "로봇의 교육적 활용을 위한 교육 프로그램 모형 개발," 한국실과교육학회지, 제16권, 제3호, pp.75-90, 2003.

[14] D. Alimisis and C. Kynigos, "Constructionism and Robotics in Education," Teacher Education on Robotic-Enhanced Constructivist Pedagogical Methods, pp.11-26, 2009.

[15] C. Chomyim, S. Chaisanit, and A. Trangansri, "Low Cost Mobile Robot Kits Design as a Teaching Tool for Education and Research," Applied Mechanics and Material, Vol.752, pp.1010-1015, 2015.

[16] Y. A. C. González and A. G. V. Muñoz-Repiso, "Educational Robotics for the Formation of Programming Skills and Computational Thinking in Childish," Computers in Education (SIIE), 2017 International Symposium on, pp.1-5, 2017.

[17] ISTE, "CT Leadership Toolkit," Available at: http://www.iste.org/docs/ctdocuments/ct-leadershipt-toolkit.pdf?sfvrsn¼4, 2015.

[18] Ö. Korkmaz, R. Çakir, and M. Y. Özden, "A Validity and Reliability Study of the Computational Thinking Scales (CTS)," Computers in Human Behavior, Vol.72, pp.558-569, 2017.

[19] 전용주, 김태영, "국내외 동향 분석을 통한 SW 교육의 이해," 한국컴퓨터학회논문지, 제18권, 제2호, pp.1-6, 2014.

[20] S. Papert, *Mindstorms(Second Addition): Children, Computers, and Powerful Ideas,* Basic Books, 1993.

[21] M. Guzdial, “Programming Environments for Novices,” Computer Science Education Research, Vol.2004, pp.127-154, 2004.

[22] 배영권, 남재원, “웹 2.0 을 활용한 로봇프로그래밍 교육이 문제해결력 신장에 미치는 영향,” 한국콘텐츠학회논문지, 제10권, 제11호, pp.468-475, 2010.

[23] 송정범, 조성환, 이태욱, “스크래치 프로그래밍 학습이 학습자의 동기와 문제해결력에 미치는 영향,” 정보교육학회논문지, 제12권, 제3호, pp.323-332, 2008.

[24] D. H. Clements and D. F. Gullo, “Effects of Computer Programming on Young Children's Cognition,” Journal of Educational Psychology, Vol.76, No.6, pp.1051-1058, 1984.

[25] F. Kalelioğlu, “A New Way of Teaching Programming Skills to K-12 Students: Code.org,” Computers in Human Behavior, Vol.52, pp.200-210, 2015.

[26] 이민영, 전석주, “엔트리와 스크래치를 활용한 초등학생의 논리적 사고력 신장에 관한 연구,” 한국초등교육, 제28권, 제1호, pp.173-185, 2017.

[27] 채재호, 배영권, 유인환, “로봇프로그래밍 학습이 초등학생의 논리적 사고력 신장에 미치는 영향,” 교원교육, 제24권, 제2호, pp.361-376, 2008.

[28] 서순식, 배영권, 양유정, 고유하, 신승기, 장의덕, 최미애, *로봇 활용 SW교육 효과성 검증 연구,* 한국교육학술정보원 연구보고, KR 2016-1, 2016.

[29] 교육과학기술부, *2009 개정 교육과정 해설,* 서울: 교육과학기술부, 2011.

[30] J. M. Wing, “Computational Thinking and Thinking about Computing,” Philosophical Transactions of the Royal Society of London A: mathematical, physical and engineering sciences, Vol.366, No.1881, pp.3717-3725, 2008.

[31] CSTA, *CSTA K-12 Computer Science Standards Revised 2011 [Eletronic version].* ACM, 2011.

[32] 최형신, “로봇활용교육의 효과성 검증을 위한 평가도구 개발: 사회·문화적 맥락 및 컴퓨팅 사고 연계,” 정보교육학회논문지, 제18권, 제4호, pp.541-548, 2014.

[33] P. Curzon, “Computational Thinking: Searching to speak” Available at: http://www.icspiazzasicilia.gov.it/wp-content/uploads/2016/04/SearchingToSpeak.pdf, 2015.

[34] V. Barr and C. Stephenson, “Bringing Computational Thinking to K-12: What is involved and what is the role of the computer science education community?,” ACM Inroads, Vol.2, No.1, pp.48-54, 2011.

[35] J. J. Lu and G. H. L. Fletcher, “Thinking about Computational Thinking,” In Proceedings of the 40th ACM Technical Symposium on Computer Science Education (SIGCSE'09), 2009.

[36] D. Barr, J. Harrison, and L. Conery, “Computational Thinking: A digital age skill for everyone,” Learning & Leading with Technology, Vol.38, No.6, pp.20-23, 2011.

[37] 김재휘, 김동호, “컴퓨팅 사고력 향상을 위한 초등 피지컬 컴퓨팅 교육과정 개발,” 정보교육학회논문지, 제20권, 제1호, pp.69-82, 2016.

[38] S. Atmatzidou and S. Demetriadis, “Advancing Students' Computational Thinking Skills Through Educational Robotics: A study on age and gender relevant differences,” Robotics and Autonomous Systems, Vol.75, pp.661-670, 2016.

[39] I. Lee, F. Martin, J. Denner, B. Coulter, W. Allan, J. Erickson, and L. Werner, “Computational Thinking for Youth in Practice,” ACM Inroads, Vol.2, No.1, pp.32-37, 2011.

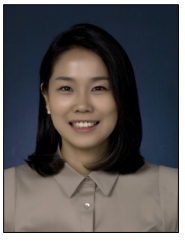
[40] 이정민, 박현경, 최형신, “로봇 활용 SW 교육이 초등학생의 컴퓨팅 사고력, 창의성, 학업흥미, 협업능력에 미치는 효과,” 정보교육학회논문지, 제22권, 제1호, pp.9-21, 2018.

[41] A. Khanlari, “Effects of Robotics on 21st Century Skills,” European Scientific Journal,

ESJ, Vol.9, No.27, pp.26-36, 2013.
[42] 전성균, 서영민, 이영준, “컴퓨터교과교육과 컴퓨터과학: 창의성과 프로그래밍 교육에 관한 고찰,” 한국컴퓨터교육학회 학술발표대회논문집, 제15권, 제1호, pp,73-77, 2011.
[43] 박경재, 이수정, “두리틀과 로봇 프로그래밍 교육이 창의성에 미치는 효과 비교 연구,” 정보교육학회논문지, 제14권, 제4호, pp.619-626, 2010.
[44] 송정범, “메타인지 전략을 활용한 게임 프로그래밍 학습이 초등학생의 문제해결력에 미치는 효과,” 교원교육, 제24권, 제4호, pp.432-447, 2008.
[45] 류충규, 이철현, “스크래치 프로그래밍이 초등영재학생들의 창의적 문제해결력에 미치는 효과,” 한국실과교육학회지, 제25권, 제1호, pp.149-169, 2012.
[46] S. Blanchard, V. Freiman, and N. Lirrete-Pitre, “Strategies sed by elementary schoolchildren solving robotics-based complex tasks: Innovative potential of technology,” Procedia-Social and Behavioral Sciences, Vol.2, No.2, pp.2851-2857, 2010.

저 자 소 개

이 정 민(Jeongmin Lee) 정회원



- 2001년 : 이화여자대학교 교육공학과(학사)
- 2003년 : 이화여자대학교 교육공학과(석사)
- 2009년 : 플로리다주립대학교 교육심리 및 교육공학(박사)
- 2010년 ~ 현재 : 이화여자대학교 교육공학과 부교수

<관심분야> : 테크놀로지기반 학습설계, SW교육, 스마트러닝, 학습정서

고 은 지(Eunji Ko) 정회원

- 2009년 : 전북대학교 수학과(학사)
- 2015년 : 이화여자대학교 교육대학원 교육공학·HRD전공(석사)
- 2015년 ~ 현재 : 이화여자대학교 교육공학과 박사과정

<관심분야> : SW교육, 교수설계, 학습정서