韓國數學敎育學會誌 시리즈 A <數學敎育>
http://dx.doi.org/10.7468/mathedu.2019.58.4.483
2019. 11. 제 58권, 제 4호, 483-505.

J. Korean Soc. Math. Ed. Ser. A:
The Mathematical Education
November. 2019, Vol. 58, No. 4, 483-505.

수학교육에서 계산적 사고(Computational Thinking)의 의미 및 연구 동향 탐색

신동조(단국대학교, 외래강사), 고상숙(단국대학교, 교수)†
†교신저자

A study on investigation about the meaning and the research trend of computational thinking(CT) in mathematics education

Shin, Dongjo(Dankook University, sdlov20@gmail.com)
Choi-Koh, Sangsook(Dankook University, sangch@dankook.ac.kr)†
†Corresponding Author

초록

세계적으로 계산적 사고를 학교 교육과정에 통합하려는 움직임이 일고 있고, 수학교과는 이러한 움직임의 핵심이 되고 있다. 본 연구에서는 Jeannette Wing의 주장과 선행연구를 바탕으로 계산적 사고와 수학적 사고 간의 관계를 분석하였고 계산적 사고를 수학교과에 통합한 국내외 연구를 종합적으로 검토하였다.

Abstract

Across the world, there is a movement to incorporate computational thinking(CT) into school curricula, and math is at the heart of this movement. This paper reviewed the meanings of CT based on the point of view of Jeanette Wing, and the trend of domestic and international studies that incorporated CT into the field of mathematics education was analyzed to provide implications for mathematics education and future research. Results indicated that the meaning of CT, defined by mainly computer educators, varied in their operationalization of CT. Although CT and mathematical thinking generally have common points that are oriented toward problem solving, there were differences in the way of abstraction that is central to the two thinking processes. The experimental studies on CT in the field of mathematics education focused mainly on the development of students' cognitive capacities and affective domains through programming(coding). Furthermore, the previous studies were mainly conducted on students in school, and the studies conducted in the context of higher education, including pre-service and in-service teachers, were insufficient. Implications for mathematics teacher educators and teacher education as well as the relationship between CT and mathematical thinking are discussed.

* 주요어 : 계산적 사고(컴퓨팅 사고), 수학적 사고, SW교육, 프로그래밍(코딩), 연구 동향
* **Key words** : computational thinking, mathematical thinking, software education, Programming(coding), research trend
* **Address**: Dept. of Mathematics Education, College of Education, Dankook University
* **ZDM Classification** : C10
* **2000 Mathematics Subject Classification** : 97-02
* **Received**: July 29, 2019 **Revised**: September 3, 2019 **Accepted**: September 30, 2019

Ⅰ. 서론

인공지능이 우리의 일상생활에 함께하면서 전 세계적으로 계산적 사고[1](computational thinking, 이하 CT)에 대한 관심이 높아지고 있고 CT를 학교 교육과정에 통합하려는 노력이 이어지고 있다. 이는 2006년 Jeannette Wing이 기술한 3쪽 분량의 짧은 기고문에서 비롯되었다고 해도 과언이 아닐 정도로 Wing이 기술한 CT는 지난 10년간 국내외 교육 연구와 현장에 많은 영향을 끼쳤다. CT는 지능정보사회로 불리는 4차 산업혁명 시대에 컴퓨터 과학자뿐만 아니라 모든 사람이 배우고 갖추어야 할 기본적인 소양으로 여겨진다(Wing, 2006; 2017). 이러한 시대적 흐름에서 현재 또는 가까운 미래에 많은 직업이 CT 역량을 갖춘 인재를 원하고 있다(Lockwood, DeJarnette, & Thomas, 2019; Wing & Stanzione, 2016). 미국 컴퓨터 학회(Association for Computing Machinery [ACM], 2014)는 2020년까지 STEM 관련 직업의 50%가 CT 역량을 요구할 것이라고 주장하였다. 또한, CT는 창의적 사고, 비판적 사고, 문제해결력 같은 4차 산업혁명 시대가 요구하는 다양한 역량과 맞물려 있으며(Ananiadou & Claro, 2009; Binkley et al., 2012), 체계적으로 사고하는 능력을 신장시키고(Kafai & Burke, 2013), 수학·과학 관련 역량 향상(Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013)에 잠재성을 지닌다. CT는 컴퓨팅[2]의 진보와 함께 과거에는 불가능했던 새로운 문제해결 전략을 사용하여 가상세계와 실제 세계에서 새로운 해법을 제시해줄 수 있다(Wing, 2006). 예를 들어, 전통적으로 컴퓨팅적 방법론을 사용하였던 물리학이나 공학뿐만 아니라 역사적으로 선형적이고 결정론적 시스템을 강조하였던 생물학과 화학에서도 최근 추계적 분석과 비선형 문제해결을 위해 컴퓨팅 방법을 사용하면서 급격한 진보를 이루는 등 현재 과학은 컴퓨팅과 더불어 르네상스 시대를 맞이하고 있다(Weintrop et al., 2016).

이러한 시대적·사회적 요구에 발맞추어 CT 역량을 갖춘 인재 육성을 위한 범정부적 투자와 노력이 이어지고 있다. 미국의 경우, 지난 3년간 국가 예산 40억 달러를 초·중등학교 컴퓨터 교육에 투자하여 모든 학생이 학교에서 컴퓨터 과학을 배울 기회를 제공하고 있다. 국가과학재단(National Science Foundation)과 국가·지역사회봉사기관(Corporation for National and Community Service)에서는 135만 달러를 투자하여 학생들에게 컴퓨터 과학을 가르치거나 CT를 통합한 교과 교육과정을 운영할 수 있도록 교사연수를 실시하고 있다(White House, 2016). 또한, CT 함양을 목표로 K-12 컴퓨터 과학 교육과정을 개발하고 이를 지속적으로 개정하고 있으며(Computer Science Teachers Association [CSTA], 2017), 덴마크, 아일랜드, 폴란드, 영국 등 유럽의 여러 국가에서도 CT 향상을 위해 프로그래밍(코딩) 수업을 국가 수준의 학교 교육과정에 통합하고 있다(Balanskat & Engelhardt, 2015). 우리나라의 경우 2015 개정 교육과정에서 자료를 수집·분석·활용하고 공학적 도구를 사용하여 효과적으로 처리할 수 있는 '정보처리 능력'을 6가지 핵심 수학 교과 역량 중 하나로 규정하였고, 2018년부터 초등학교와 중학교에 CT 강화를 위한 소프트웨어교육을 의무화하고 있다(Ministry of Education, 2015a).

CT에 관한 연구도 다양한 분야에서 활발히 이루어지고 있다. 이는 CT가 컴퓨터 과학 분야에 국한된 사고가 아니기 때문이다. CT는 수학, 과학, 기술, 공학, 예술로 대표되는 STEAM 교육에서뿐만 아니라 의학, 경제, 법학, 사회과학, 언론학 등 다양한 분야에 영향을 미치고 있으며, CT를 통합한 융합 연구(예를 들어, 디지털 고고학, 디지털 인문학, 컴퓨터 금융학)는 우리 사회 전반에 새로운 통찰을 가져다주고 있다(Wing, 2017). 수학교육에서도 CT 통합의 필요성이 지속적으로 제기되고 있다(Chang, 2017; Shim & Shim, 2018). 이는 CT의 특성이 수학적 사고(mathematical thinking: MT)의 특성과 비슷한 맥락에서 정의되고 있기 때문이다(diSessa, 2018; National Research Council [NRC], 2010; Sneider, Stephenson, Schafer, & Flick, 2014; Wing, 2006; 2008). 나아가, 4차 산업혁명 시대 컴퓨팅을 이용한 문제해결능력 함양은 수학교육의 중요한 역할 중 하나로 부각되고 있다(Kim, Seo, & Cho, 2018; Shim & Shim, 2018).

---

[1] CT를 국내에서 컴퓨팅 사고 또는 계산적 사고라고 불러지고 있는데, 컴퓨팅은 computing 영어단어를 소리나는대로 한글화하는 관행에 따른 것으로 보이며 이는 computational이라는 원단어의 의미가 희석된 것으로 인식할 수 있어 계산적 사고로 선도해야 할 필요가 제기된다. 이는 초기에 CT가 컴퓨터 환경에서 가능한 사고임을 뜻하기 위해 계산적보다 컴퓨팅으로 소개된 것이라 판단된다.

[2] 본 연구에서 컴퓨팅(computing)은 컴퓨터의 활용을 의미한다.

Weintrop 외(2016)는 수학·과학 교과에서 CT 통합의 필요성을 3가지로 기술하였다. 첫째, 앞서 기술하였듯이 시대적으로 수학·과학 관련 직업은 CT 역량을 갖춘 인재를 필요로 한다. 지난 20년간 다수의 수학·과학 관련 분야는 컴퓨팅의 통합으로 성장했고, 생체정보학(Bioinformatics), 전산통계학(Computational Statistics), 계량화학(Chemometrics), 신경정보학(Neuroinformatics)과 같은 분야의 성장을 가능하게 했다. 또한, STEM 분야의 성장과 STEM에서 컴퓨팅 역할의 중요성이 이와 맥락을 같이한다. 둘째, 교육학적 관점에서 수학·과학 교과와 CT는 상호보완적 관계를 가진다. 먼저, 계산적 도구와 기술의 전략적 사용은 수학·과학 개념이해를 도울 수 있다. 반대로, CT 교육에서 수학·과학 관련 내용과 문항 구성은 CT를 적용할 수 있는 의미 있는 맥락을 제공할 수 있다. 이는 CT를 실생활에 적용하고 다른 분야에 통합한다는 관점에서 컴퓨터 과학 내에서 CT를 가르치는 것보다 효과적이다. 마지막으로, 수학·과학 교과에 CT 통합은 전통적으로 컴퓨터 관련 분야에서 배제되었던 여성과 소외계층에게 자연스럽게 교육의 기회를 제공할 수 있다. 요약하면, 수학 교과에 CT 통합은 직업적 역량 강화, 수학적 이해 및 CT 향상, 교육의 공정성(equity)에 기여할 수 있다(Weintrop et al., 2016).

국내외 CT와 관련한 문헌연구와 동향분석은 지속적으로 진행되었다. 하지만, 대부분 컴퓨터 과학 관련 CT 선행연구 분석(Grover & Pea, 2013; Shute, Sun, & Asbell-Clarke, 2017), 프로그래밍(코딩) 관련 CT 선행연구 분석(Lye & Koh, 2014), 소프트웨어교육 관련 연구 동향분석(Hwang & Hwang, 2017; Lee, 2018) 등을 위주로 수행되었다. Chang(2017)은 수학교육, 컴퓨팅, CT와의 관계를 고찰하면서 수학 교과에서 CT 통합 교육 가능성을 시사하였지만, 수학적 사고력과 수학적 맥락에서 CT에 관한 선행연구를 종합적으로 검토한 연구는 드물다. CT의 적용범위가 광범위한 만큼 수학 교과 관련 CT 연구가 어떻게 이루어지고 있는지 주목할 필요가 있다. 특히, 대부분의 CT 연구가 Wing의 2006년 기고문으로부터 시작한다는 점에서 Wing이 말하는 CT의 개념과 특성에 대해 살펴볼 필요가 있다.[3)]

따라서 본 연구에서는 첫째, 2006년 Wing의 기고문을 시작으로 후속 논문(Wing, 2008; 2014; 2017; Wing & Stanzione, 2016)을 종합적으로 검토하여 CT의 개념과 특성에 대한 Wing의 주장을 명확히 한 후, CT를 수학교과와 관련하여 기술한 선행연구 결과를 분석하여 정의된 CT의 이론적 틀을 통해 수학교육 맥락에서 CT의 의미를 규명한다. 둘째, CT를 통합한 수학교육 관련 연구에서 이들 연구가 학생의 인지적 영역과 정의적 영역에 미친 영향과 이들 연구에서 사용된 처치 방안의 효과를 조사한다. 이로써 본 연구결과는 MT와 구별되는 CT의 개념을 명확히 규명하고 이를 수학교육에서 의미 있게 통합할 수 있는 방안을 살펴봄으로써 앞으로 이뤄질 연구 방향에 대한 중요한 시사점을 제공할 것으로 보인다.

Ⅱ. 이론적 배경

Wing에 따르면 CT는 “컴퓨터 과학자처럼 사고하는 것”(Wing, 2006; 2017; Wing & Stanzione, 2016)으로 “컴퓨터 과학의 근본적인 개념을 사용하여 문제를 해결하고, 시스템을 설계하고, 인간의 행동을 이해하는 것”을 의미한다(Wing, 2006; 2008). Wing의 최근 논문을 살펴보면 “CT란 컴퓨터(인간 또는 기계)가 효과적으로 수행할 수 있는 방식으로 문제를 공식화(formulating)하고 해법을 표현하는 것(expressing)과 관련된 사고과정”으로 정의된다(Wing, 2017, p. 8). 여기서, ‘효과적’이란 컴퓨터를 통해 ‘계산 가능한(computable)’을 의미하고, ‘표현하는 것’이란 해법을 다른 사람이나 기계와 의사소통하기 위해 언어적 표상을 만드는 것을 의미한다. 일반적으로 컴퓨터와 의사소통하기 위한 언어적 표상은 프로그래밍(코딩) 언어를 의미하지만, 인간은 컴퓨터 프로그래밍(코딩) 언어로 의사소통하지 않기 때문에 CT를 위해 반드시 컴퓨터가 있어야 하는 것은 아니다(Wing, 2008; 2017). 2006년과 2008년 Wing이 정의한 CT에는 ‘문제해결’을 CT의 핵심으로 포함했지만, 이후 CT를 “문제해결에 관한 것뿐만 아니라 문제의 공식화(problem formulation)에 관한 것”으로 CT의 범위를 보다 포괄적으로 정의하고자 했다(Wing, 2017, p. 8).

3) CT와 관련된 논의는 Wing 이전에 많은 학자(예: diSessa, 2000; Papert, 1980)에 의해 다루어졌지만, Wing의 2006년 기고문이 CT의 대중화에 지대한 영향을 끼쳤다는 점에서 본 연구는 Wing의 정의로부터 시작한다.

Wing이 주장하는 CT의 핵심요소는 추상화(abstraction)와 자동화(automation)이고, 그중 추상화를 강조한다. Wing은 2006년 그녀의 기고문에서는 추상화를 간단하게 언급하였다. 즉, CT는 다양한 수준의 추상화를 요구하는 사고이며, 복잡한 문제나 시스템을 설계할 때 추상화와 분해(decomposition)를 사용하는 것이라고 하였다. 여기서 추상화는 분해와 명확한 구분 없이 사용되었는데, 문제를 해결 가능한 단위로 만들기 위해 문제에 관련된 속성을 모델링하거나 문제에 적합한 표상을 선택하는 것을 의미한다. Wing은 2008년부터 추상화를 CT의 핵심으로 강조하였다. 2008년 그녀의 논문에서 "CT의 본질(essence)이 추상화"라고 하였으며(Wing, 2008, p. 3717), 2017년 논문에서도 "컴퓨터 과학은 추상화의 자동화이며, CT에서 가장 중요하고 높은 수준의 사고과정이 추상화 과정"이기에 추상화가 CT의 핵심(key)이라고 기술하였다(Wing, 2017, p. 8). Wing(2008; 2017)이 말하는 추상화란 여러 대상 중 관련 없는 세부사항은 제거하고, 공통으로 나타나는 핵심 속성을 파악하는 것이다. 즉, 그녀가 말한 "해결 가능한 단위로 만드는 과정"을 의미한다(Wing, 2006, p. 33). 이러한 추상화는 컴퓨팅에 관한 정신적 도구로써 추상화의 힘은 기계적 도구의 힘에 의해 증폭될 수 있다. 이러한 기계적 도구의 힘이 자동화이다. Wing의 자동화는 추상화된 개념들을 해석할 수 있도록 컴퓨터 등을 통해 작동시키는 것을 의미한다. 종합하면, Wing의 CT는 효과적인 문제해결을 위해 문제의 핵심을 추상화하고 이를 자동화하여 표현하는 데 필요한 사고과정을 의미한다.

Ⅲ. 연구 방법

1. 분석대상

본 연구는 국내외에서 수행된 CT 관련 연구 동향을 수학교육을 중심으로 분석하고자 하였다. 먼저, 자료수집 기간은 Wing의 2006년 기고문을 시작으로 본 연구가 수행된 2019년 5월까지로 선정했다. 국외논문을 수집하기 위해 ERIC(Education Resources Information Center, eric.ed.gov)에서 'mathematics education'을 주제어로 'computational thinking', 'programming', 'coding'을 교차 입력하였다. 추가적으로, Williams, Leatham(2017)의 국외 수학교육 학술지 평가에 관한 연구에서 가장 좋은 등급을 받은 7개 학술지[4]에서 CT관련 논문을 포함하여 총 335편의 국외논문을 수집하였다. 국내논문 수집을 위해 학술연구정보서비스(riss.kr), 한국학술지인용색인(kci.go.kr), 한국학술정보(kiss.kstudy.com)에서 '수학'을 중심으로 '컴퓨팅 사고', '계산적 사고', '프로그래밍'과 '코딩'을 교차 검색하여 325편을 찾아내었다. 이들 논문을 대상으로 제목, 키워드, 초록을 검토하여 수학, 수학적 사고, 수학적 문제해결력과 관련 없다고 사료되는 논문은 1차 분석에서 제외하였다. 이후, 논문 내용을 전체적으로 검토하여 위에 제시된 포함 기준과 관련 없었던 논문은 최종 분석에서 제외하였다. 마지막으로, 선정된 논문들에서 참고했던 문헌 중 본 연구의 주제와 관련되는 논문들을 추가하여 최종 97편의 논문(국외논문: 52편, 국내논문: 45편)이 분석 자료로 선택되었다.

2. 분석방법

본 연구에서는 97편의 수학교과 관련 CT 선행연구를 크게 3가지 주제로 검토하였다. 첫째, CT와 수학적 사고와의 관계를 기술한 연구를 분석하여 두 가지 사고 간의 공통점과 차이점을 살펴보았다. 이 과정에서 CT와 수학적 사고와의 관계를 핵심 주제로 연구한 논문이 없었기 때문에 두 사고의 관계를 언급한 모든 논문을 이 범주에 포함시켜 분석하였다. 둘째, 수학적 맥락에서 CT에 관한 이론적 틀을 제공한 연구를 분석하였다. 셋째, CT를 통합한 수학교육이 학습자의 인지적·정의적 영역에 미친 영향과 이를 위한 교수학적 처치 방안에 대해 검토하였다. 교수학적 처치 방안은 프로그래밍(코딩) 도구 활용, 로봇 활용, 프로그래밍(코딩)이 아닌 컴퓨팅 도구 활용, 컴퓨팅 도구를 사용하지 않은 연구로 분류하였고, 연구 대상, 처치 기간, 연구 방법, 연구 결과 등으로 세분화하여 분석하였다.

---

[4] 7개의 학술지는 다음과 같다: <Journal for Research in Mathematics Education>, <Educational Studies in Mathematics>, <The Journal of Mathematical Behavior>, <Zentralblatt für Didaktik der Mathematik(ZDM)>, <Journal of Mathematics Teacher Education>, <Mathematical Thinking and Learning>, <For the Learning of Mathematics>

Ⅳ. 결과 분석 및 논의

1. 결과

1) 계산적 사고(CT)와 수학적 사고(MT)의 관계

수학적 사고와 CT의 관계에 대한 논의는 학자들 사이에 논쟁을 불러일으키고 있고, 여전히 명확한 관계를 규명하지 못하고 있다(Barr, Harrison, & Conery, 2011; Brennan & Resnick, 2012; Grover & Pea, 2013; Shute et al., 2017). 본 절에서는 선행연구에서 수학적 사고와 CT와의 관계를 어떻게 기술했는지 살펴보고 이를 종합적으로 분석해보고자 한다. 이를 위해 먼저 Wing의 논고에서 수학 또는 수학적 사고를 어떻게 기술하였는지 살펴볼 필요가 있다.

Wing에 따르면 모든 과학이 수학에 기초하듯 컴퓨터 과학 역시 수학에 기초하기 때문에 CT는 본연적으로 수학적 사고에 의존한다고 주장하였다(Wing, 2006). 또한, Wing은 CT가 일반적으로 문제해결에 접근하는 방식에서 수학적 사고와 공통점을 가지고 있다고 하였다(Wing, 2008). Wing은 CT와 수학적 사고와의 차이를 추상화 방식에서 설명하였다. 구체적으로 컴퓨터 과학에서의 추상화는 수학과 물리학에서의 추상화보다 두 가지 측면에서 더 복잡한 경향이 있다고 하였다. 첫째, 컴퓨터 과학에서 추상화는 수와 집합과 같은 대상을 간단한 대수적 연산을 통해 추상화하지 않는다. 예컨대, 컴퓨터 과학에서는 수학에서 두 정수를 더하는 것처럼 두 개의 알고리즘을 더하는 방식으로 추상화하지 않는다(Wing, 2008). 둘째, 컴퓨터 과학에서의 추상화는 물리적인 제약 속에서 수행되기 때문에 극단적 상황이나 오류 발생 등(예컨대, 컴파일 과정에서 나타나지 않았지만 프로그래밍을 실행할 때 발생하는 오류)를 고려해야 한다는 점에서 수학에서의 추상화보다 복잡한 성격을 가진다(Wing, 2008).

요약하면, Wing의 지난 10년간 논문에서 CT와 수학적 사고는 궁극적으로 문제해결을 지향한다는 점과 CT가 본연적으로 수학적 사고에 의존한다는 것에서 유사하지만 추상화 방식에서 차이가 있었다. 하지만, Wing은 CT와 수학적 사고의 관계에 대해서 구체적으로 기술하지 않았다는 점에서 두 사고 간의 관계를 서술한 후속연구를 살펴볼 필요가 있다.

국제교육공학회(International Society for Technology in Education, ISTE)와 컴퓨터과학교사협의회(CSTA)에서 주간한 세미나에서는 CT와 비판적·수학적 사고와의 차이에 대해 아래와 같이 설명하였다(Barr et al., 2011, p. 23).

- CT는 다른 분야에 통합되어 사용되었을 때 새로운 방식으로 효과적인 문제해결의 기초를 제공한다는 점에서 비판적·수학적 사고와 구별되는 독특한 사고 기술의 조합이다.
- CT는 비판적·수학적 사고보다 도구(tool) 지향적이다.
- CT는 이전 시대에는 실행 불가능했지만 현재 빠른 속도의 자동화가 가능한 도구를 사용하여 추측, 반복, 시행착오와 같은 익숙한 문제해결 전략을 사용한다.

다시 말해, CT는 반복, 추측, 시행착오라는 다소 단순한 문제해결 전략에 의존하지만, 도구를 통한 자동화로 다른 교과에 적용하였을 때 새로운 통찰을 줄 수 있다는 점에서 수학적 사고와 구별되는 사고과정이다. 하지만, 이 세미나는 역시 주로 교육공학과 컴퓨터 교육 전문가들이 참여하였기 때문에 수학적 사고에 대해 심도 있는 논의가 이루어지지 않았던 것으로 사료된다. 다시 말해, Barr 외(2011)의 논문에서는 수학적 사고의 특성에 관한 어떠한 설명도 제시하지 않았다.

전미연구위원회(NRC)에 따르면, CT와 수학적 사고는 추상화와 추상화를 통해 단순화된 모델에 관한 추론을 기반으로 한다는 점에서 공통점을 가진다(NRC, 2010). 또한, CT와 수학적 사고는 고유의 언어(프로그래밍 언어와 수학기호)를 통해 대상을 표현한다는 점에서 비슷하다. 하지만, 수학적 사고는 추상적인 방법론(abstract methodology)이라기보다 추상적인 구조(abstract structure)에 관한 사고에 가깝다. 예컨대, 프로그래밍 언어는 특정한 대상을 구현하도록 하는 방법에 관한 언어이고 이에 명확한 기술이 뒷받침되지만, 수학적 언어는 그 대상 자체의 구조를 기술하는 언어에 가깝다는 점에서 프로그래밍 언어와는 구별된다. 이는 컴퓨터 과학에서의 추상화와 수학적 추상화가 다르다는 관점에서 Wing(2008)의 의견과 일치하지만, 두 유형의 추상화에 대한 차이점을 서술하는 방식은 상이하다.

수학적 추상화와 컴퓨터 과학에서의 추상화의 차이는 Andrea diSessa를 통해 보다 심도 있게 논의되었다.

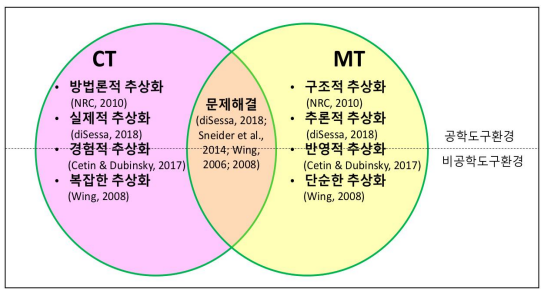
diSessa(2018)는 추상화가 수학, 물리학, 컴퓨터 과학에서 각기 다르게 정의되어야 한다고 주장하였다. 먼저 수학적 추상화는 추론적 추상화(inferential abstraction)라고 할 수 있는데, 이는 속성들로 이루어진 작은 집합이 수학적 추상화를 통해 실체(entities)를 정의하는 방식으로 개념적 세계를 만들고, 결과적으로 기본적인 아이디어와 추론(증명)으로부터 다른 아이디어(이론)를 아우르는 집합체인 견고한 추론적 구조를 만들어낸다. 반면 컴퓨터 과학에서의 추상화는 실제적 추상화(practical abstraction)라 할 수 있는데, 예컨대 프로그래밍(코딩)을 통한 시스템 구현(implementation)을 위해 관련 없는 세부사항들은 제거하고 필수적인 속성만 도출하는 것을 의미한다. 결국, diSessa가 말하는 수학적 추상화는 추론적 구조를 만들어내기 위한 과정이라는 점에서 NRC(2010)에서 설명한 구조적 추상화와 유사하며, 컴퓨터 과학에서 추상화는 실제적 구현을 위해 핵심 사항을 도출하는 과정이라는 점에서 방법적 추상화(NRC, 2010)와 유사하다.

Cetin, Dubinsky(2017)는 CT에서 말하는 추상화를 Piaget의 추상화와 관련하여 설명하였다. Cetin, Dubinsky에 따르면, Wing이 말하는 추상화(세부사항들은 무시하고 필수적인 속성만 추출하는 것)는 Piaget의 경험적 추상화(empirical abstraction)와 관련된다. 예컨대, 아이들은 여러 종류의 '개'를 보고 세부사항들은 무시하고 공통적인 속성(4개의 다리, 꼬리 흔들기, 짖는 행위 등)을 추출하면서 개에 대한 개념을 경험적으로 추상화한다. 비록 많은 수학적 개념이 공통적인 속성을 추출하는 과정으로 형성되지만, 수학적 사고의 기본이 되고 주체의 행동에 대한 조정과 관련되는 반영적 추상화(reflective abstraction)와는 구별된다고 하였다.

diSessa(2018)는 앞서 살펴본 것과 같이 추상화 과정에서 CT와 수학적 사고와의 차이가 있지만, Wing이 기술한 CT가 George Pólya가 말한 수학에서의 문제해결(problem solving)과 유사하다고 하였다. 예를 들어, Wing(2006)이 주장한 문제를 해결 가능한 단위로 분해한다는 것은 컴퓨터 프로그래밍(코딩)에서 주요한 내용인데 Pólya는 1945년 그의 책 How to solve it에서 본질적으로 같은 아이디어를 서술하였다. Pólya는 문제해결을 위해 '그림(figure)을 그리는 것'을 강조하였는데, Wing은 이를 표상(representation)이란 단어를 사용하여 보다 일반적으로 기술하였지만 핵심 아이디어는 Pólya의 것과 같다. 또한, Pólya의 '일반화'는 Wing의 '추상화'의 개념과 같지는 않지만 비슷한 속성을 가진다. Wing(2014)은 추상화를 통한 CT의 발달은 다른 영역으로 '전이'(transfer)될 수 있다고 주장하였는데 이를 통해 Wing이 가지는 추상화의 과정은 일반화의 속성을 가진다고 할 수 있다. 마지막으로, Pólya는 문제해결을 위한 계획을 세우고 이를 효과적으로 수행하는 것을 강조했는데 Wing(2017) 역시 CT를 정의하기 위해 비슷한 맥락을 사용하였다. 정리하면, diSessa는 CT와 수학적 사고와의 연관성을 문제해결로 보았고, 수학적 추상화와 컴퓨터 과학에서의 추상화를 구분해야 할 필요가 있음을 시사하였다.

Sneider 외(2014)는 CT와 수학적 사고의 관계를 벤다이어그램을 통해 보다 명확하게 구분하고자 하였다. Sneider 외에 따르면 CT와 수학적 사고는 공통적으로 문제해결, 모델링, 데이터 분석 및 해석, 확률과 통계에 대한 지식을 포함하고, CT의 고유한 영역으로 시뮬레이션, 데이터 마이닝, 네트워킹, 자동화된 자료 수집, 게임밍(gaming), 알고리즘적 사고, 로보틱스(robotics), 프로그래밍(코딩)이, 수학적 사고의 고유한 영역으로 셈하기, 산술, 대수, 기하, 미적분, 집합론, 위상수학이 있다고 하였다. 하지만, Burton(1984)은 수학적 사고란 그 사고가 수학(교과 영역)에 대해 사고한다는 의미보다 그 사고과정이 의존하는 연산이 수학적 연산이기에 수학적 사고라고 하였다. 따라서 Sneider 외가 서술한 CT와 수학적 사고의 관계는 두 종류의 사고과정이 어떠한 관계가 있는지를 규명했다기보다 수학 교과와 컴퓨터 과학 교과에서 가르치는 영역 사이의 관계에 주목했다고 볼 수 있다.

종합하면, 선행연구에서는 CT와 수학적 사고가 대체로 문제해결을 지향한다는 점과 문제해결 방식에서 유사한 사고과정이라고 주장하였다. 반면 CT와 수학적 사고는 추상화 방식에서 구별되어야 한다고 하였다. 두 사고 간의 추상화 방식을 나타내는 용어는 연구자 사이에서 상이했지만 CT에서의 추상화는 실제 프로그램 구현을 위해 본질적인 속성을 추출하는 방법적, 실제적, 경험적 추상화를 의미하는 반면 수학적 추상화는 개념에 대한 구조적, 추론적, 반영적 추상화로 보다 복잡한 추상화 과정을 의미했고 이는 Wing이 구별하는 추상화와는 대조되었다([Fig. 1] 참고).



[Fig. 1] Similarities and differences between CT and mathematical thinking(MT)

2) 수학교육에서 계산적 사고에 관한 이론적 틀

지난 10년간 CT에 대한 높은 관심으로 CT를 이해하고 분석하기 위한 다양한 이론적 틀이 제안되었다(Barr & Stephenson, 2011; Brennan & Resnick, 2012; Gouws, Bradshaw, & Wentworth, 2013; Perez, 2018; Weintrop et al., 2016). 본 절에서는 수학 교과 및 수학적 맥락에서 CT에 관한 이론적 틀을 제시한 연구를 살펴보고 이러한 이론적 틀의 사용가능성과 한계점에 대해 논의하고자 한다.

CT를 K-12 교육과정에 통합하려는 노력의 일환으로 CSTA와 ISTE는 K-12에서 CT의 핵심요소를 자료수집, 자료분석, 자료표현, 문제 분해, 추상화, 알고리즘 및 절차, 자동화, 병렬화(parallelization), 시뮬레이션으로 구분하고 이를 컴퓨터 과학, 수학, 과학, 사회학, 언어학의 관점으로 설명하였다(Barr & Stephenson, 2011, [Table 1] 참고). 비록 Barr, Stephenson(2011)은 CT 구성요소를 수학교과 내에서 예를 통해 보여줬지만, 이러한 예는 단편적이고 피상적으로 다루어져서 연구자나 교사가 실제로 사용하기 힘들다. 따라서, 이를 수학 교과에 적용하기 위해서는 교과에 타당한 CT의 정의와 구체적인 설명이 요구된다(Shute et al., 2017).

Weintrop 외(2016)는 CT가 광범위하게 사용되기 때문에 수학·과학 교과에 적합한 CT의 정의와 이론적 틀을 제안하였다. 먼저, CT에 관한 문헌 조사를 통해 반복적으로 중요하게 거론된 CT 핵심 기술과 역량을 10개로 정리한 뒤, STEM 프로젝트에서 CT 관련 연구를 수행 중인 대학원생과 현직 교사로부터 34개의 수학·과학 수업지도안(수학 수업지도안 11개 포함)을 수집하여 CT 관련 실천 행위(practices)를 분석하였다. 이러한 과정을 통해 수학·과학에서 CT 관련 4개의 실천 행위(자료, 모델링과 시뮬레이션, 계산적 문제해결, 시스템적 사고)와 각 실천 행

[Table 1] Barr, Stephenson's(2011, p. 52) framework for CT in computer science and mathematics

	Computer Science	Mathematics
Data collection	Find a data source for a problem area	Find a data source for a problem area, for example, flipping coins or throwing dice
Data analysis	Write a program to do basic statistical calculations on a set of data	Count occurrences of flips, dice throws and analyzing results
Data representation	Use data structures such as array, linked list, stack, queue, graph, hash table, etc.	Use histogram, pie chart, bar chart to represent data; use sets, lists, graphs, etc. to contain data
Problem decomposition	Define objects and methods; define main and functions	Apply order of operations in an expression
Abstraction	Use procedures to encapsulate a set of often repeated commands that perform a function; use conditionals, loops, recursion, etc.	Use variables in algebra; identify essential facts in a word problem; study functions in algebra compared to functions in programming; Use iteration to solve word problems
Algorithms & procedures	Study classic algorithms; implement an algorithm for a problem area	Do long division, factoring; do carries in addition or subtraction
Automation		Use tools such as: geometer sketch pad; star logo; python code snippets
Parallelization	Threading, pipelining, dividing up data or task in such a way to be processed in parallel	Solve linear systems; do matrix multiplication
Simulation	Algorithm animation, parameter sweeping	Graph a function in a Cartesian plane and modify values of the variables

[Table 2] Weintrop et al.'s(2016, p. 135) framework for CT practices in mathematics and science education

Data Practices	Modeling & Simulation Practices	Computational Problem Solving Practices	Systems Thinking Practices
Collecting data	Using computational models to understand a concept	Preparing problems for computational solutions	Investigating a complex system as a whole
Creating data	Using computational models to find and test solutions	Programming	Understanding the relationships within a system
Manipulating data	Assessing computational models	Choosing effective computational tools	Thinking in levels
Analyzing data	Designing computational models	Assessing different approaches/solutions to a problem	Communicating information about a system
Visualizing data	Constructing computational models	Developing modular computational solutions	Defining systems and managing complexity
		Creating computational abstractions	
		Troubleshooting and debugging	

위별 5~7개의 하위요소로 구성된 틀을 제안했다([Table 2] 참고).

Weintrop 외(2016)가 제안한 CT 관련 실천 행위와 하위요소 중 설명이 필요한 부분에 대해 선택적으로 살펴보기로 하자. 먼저, '자료'에 관한 실천 행위에서 '자료 생성'은 탐구하고자 하는 주제가 쉽게 관찰·측정할 수 없을 때(예를 들어, 은하계의 진화를 탐구) 계산적 절차를 세우고 이를 시뮬레이션을 통해 구현하여 현상을 이해하는 목적으로 자료를 생성하는 것이다. '자료 처리'는 컴퓨팅 도구를 사용하여 자료를 분류, 필터링, 정규화, 이질적인 자료를 결합하는 등 탐구 주제에 대한 이해를 높이기 위해 자료를 유용한 형태로 재구성하는 것을 말한다. '모델링과 시뮬레이션' 실천 행위에서 계산적 모델은 컴퓨터로 시뮬레이션 될 수 있는 현상에 관한 동적 표상(non-static)을 의미한다. 여기서 '계산적 모델 설계'는 모델의 구성요소를 정의, 요소들이 상호작용하는 방식을 설명, 모델에 의해 생성될 수 있는 자료를 결정, 모델 내 가정을 기술, 모델로부터 어떠한 결론을 끌어낼 수 있을지 이해하는 과정을 통해 새로운 계산적 모델을 결정하는 것이다. 이에 반해, '계산적 모델 구성'은 주어진 모델링 언어나 도구에서 새로운 모델을 만들거나 이미 존재하는 모델을 확장함으로써 새로운 계산적 모델을 실행하는 것이다.

'계산적 문제해결' 실천 행위에서 문제해결이란 컴퓨터 과학 분야의 개념과 컴퓨팅 도구 사용했을 때 특히 효과적인 문제해결을 의미한다. 계산적 문제해결 과정이 복잡할 경우 이를 독립적이고 재사용 가능한 작은 단위로 분해하여 각 단위에 대한 문제를 해결하는 것이 효과적이다. 이때 작은 단위로 분해하는 과정을 모듈화라고 하는데 '모듈화된 계산적 해법 개발'이란 이렇게 모듈화된 요소들의 해법을 개발하고 나아가 이 요소들을 새로운 문제에 사용하는 것을 포함한다. 계산적 문제해결 과정에서 '계산적 추상화'는 문제의 중요하지 않은 속성을 숨기고 중요한 속성을 강조하여 일반적으로 표현하는 것이다. 이를 통해 표면적으로는 다르지만 구조적 유사성을 가지는 다양한 문제를 해결할 수 있다. '시스템 사고'는 하나의 시스템[5]과 그것의 부분 요소들이 전체적으로 어떻게 상호작용하는지를 포괄적인 안목으로 검토하는 것이다. CT에서 시스템 사고는 모델, 시뮬레이션과 같은 계산적 도구를 사용하여 시스템을 하나의 전체적 실체로 보는 것('복잡한 시스템을 전체적으로 검토')과 어떻게 시스템 내의 요소들이 유기적으로 상호작용하는지를 보는 것('시스템 내 부분 요소 간의 관계 이해')을 의미한다. 또한, 시스템 사고는 시스템을 미시적인 관점에서부터 거시적인 관점까지 '다양한 수준에서 생각'하는 것을 의미한다.

5) 여기서 시스템의 정의는 광범위한데 한 명의 교사와 학생들로 이루어진 교실도 하나의 작은 시스템이라 할 수 있고, 은하수의 별 또는 인간의 유전자 집단도 하나의 시스템이라 할 수 있다.

Weintrop 외(2016)는 수학·과학 교과 내 CT 실천 행위에 관한 이론적 틀을 제공하였다는 점에서 시사하는 바가 크다. 나아가 CT의 실천 행위와 발달은 학생이 가지고 있는 인지적인 측면을 이용하려는 의지와 기질 등의 정의적인 측면에 영향을 받는다(Perez, 2018). Barr, Stephenson(2011)은 CT에 해당되는 가치, 동기, 감정, 고정관념, 태도를 설명하기 위한 CT 기질(disposition)과 성향(pre-disposition)을 6가지로 범주화했다: (a) 복잡성을 다룰 수 있는 자신감, (b) 어려운 문제를 해결하는데 있어서의 끈기(persistence), (c) 모호함(ambiguity)을 다루는 능력, (d) 개방형 문제를 다루는 능력, (e) 다른 사람과 공동의 목적이나 해법을 위해 차이를 없애는 것, (f) 다른 사람의 장점과 단점을 아는 것. CT 기질에 대한 Barr, Stephenson의 분류는 Perez(2018) 연구의 토대가 되었다. Perez는 수학교육에서 CT의 역할은 수학이 다른 교과(과학, 공학, 기술 등)와의 융합을 용이하게 하는 것이라 주장했다. 다시 말해, CT는 학생들에게 수학적 아이디어를 광범위하게 적용할 수 있는 기회를 제공함으로써 수학적 사고를 보완하는 관계 속에서 통합된다고 하였다. 이는 수학적 사고의 발달을 강조하는 것이 종종 학생의 수학적 개념이해를 높이는 것(inward-oriented)을 의미하는데, CT는 이러한 방향성을 외적으로 돌려 다양한 분야에 수학적 개념이 적용 가능함을 인식하게 한다는 것(outward-oriented)이다. Perez는 CT 통합 수학교육이 학생의 문제해결과 수학적 추론에 의미 있게 다가가기 위해서는 CT에 관한 기질을 발달시키는 것이 필요하다고 주장하면서 수학교육에서 CT 기질에 관한 이론적 틀을 제시하였다([Table 3] 참고). Perez(2018, pp. 443-450)의 이론적 틀은 Barr, Stephenson(2011)에서 간단하게 기술되었던 CT 기질에 관한 범주를 3개의 요소로 통합하였고, 각 기질에 대한 성향(inclinations), 민감성(sensitivities), 능력(abilities)과 서로 다른 발달 수준과 예시를 구체적으로 제시하였다.

살펴본 바와 같이 CT에 대한 연구가 아직 초기 단계인 만큼 수학교과에서 CT에 관한 이론적 틀을 제시한 연구도 드물게 수행되었다. 그럼에도 불구하고 본 연구에서 검토된 CT의 실천 행위(Weintrop et al., 2016)와 기질(Perez, 2018)에 관한 이론적 틀은 후속연구에서 상호보완적으로 사용될 수 있다. 즉, 수학 교과 맥락에서 검토된 CT에 관한 이론적 틀은 교사 또는 연구자가 학생의 CT

[Table 3] Perez's (2018) theoretical framework for CT disposition in mathematics education

Disposition	Definition	Levels	
Tolerance for ambiguity (p. 444)	A tendency to experience ambiguous situations or stimuli as enriching and engaging	High	Learner demonstrates a willingness to engage with ambiguous situations/stimuli as valuable opportunities for discovery of that which she or he does not yet know.
		Developing	Learner may exhibit negativity in the face of ambiguous stimuli and situations and/or may seek to avoid engaging with them.
Persistence on difficult problems (p. 447)	A tendency to continue working or to maintain effort when dealing with a challenging task	High	Learner encounters challenge in the task and continues to engage with the challenge until it is resolved or until there is an outside constraint.
		Developing	Learner encounters challenge in the task and continues for some time and/or makes one or more attempts to overcome the difficulty and complete the task.
		No	Learner does not encounter challenge and/or does not engage with the challenge that inheres in the task.
Collaboration with others (p. 449)	A tendency to coordinate effort and negotiate meaning with peers to accomplish a shared goal	High	Learner will see peers as possessing unique perspectives that can be brought meaningfully to bear on a shared task or goal through a process of negotiation and exchange. This learner will take advantage of opportunities for clarification and questioning and display a willingness to pursue a course of action that may differ from what the learner would have imagined individually.
		Developing	Learner may avoid interaction altogether or limit interaction to instrumental approaches to others; that is, the learner may see others merely as a "means to an end" rather than as co-participants in a process or co-creators of meaning.

능력과 기질을 설명할 수 있는 틀로 사용할 수 있으며, CT 능력과 기질 발달을 위한 교수학적 설계를 위해 활용될 수 있다는 점에서 시사하는 바가 크다.

3) 교수학적 처치 방안

CT 통합 수학 교수학적 설계를 통해 학습자의 인지적·정의적 영역에서의 발달을 조사한 연구로는 프로그래밍(코딩) 도구를 사용한 연구, 로봇을 활용한 연구, 프로그래밍(코딩)이 아닌 컴퓨팅 도구를 사용한 연구, 컴퓨팅 도구를 사용하지 않은 연구가 있었다.

(1) 프로그래밍(코딩) 도구를 사용한 연구

다수의 CT 통합 수학교육 연구에서는 프로그래밍(코딩) 도구를 사용하여 학생의 인지적·정의적 영역에서의 변화를 조사하였다([Table 4] 참고). 프로그래밍(코딩) 도구에는 스크래치와 같이 드래그엔 드롭(drag and drop)방식의 블록기반 프로그래밍 도구와 파이썬(Python), 로고(LOGO)와 같은 언어기반 프로그래밍(코딩) 도구가 사용되었다. 예를 들어, Calao, Moreno-Leon, Correa, Robles(2015)는 수학에서의 프로그래밍(코딩) 사용이 수학적 역량(현실 세계를 모델링하는 능력, 추론능력, 문제형성 및 해결능력, 절차와 알고리즘의 비교 및 실행능력)에 미치는 영향을 조사하였다. 이를 위해, 42명의 6학년 학생을 실험집단과 통제집단으로 나누고, 실험집단에서는 스크래치를 사용한 수학 코딩 수업을, 통제집단에서는 전통적인 방식의 수학 수업을 실시하였다. 스크래치 기반 수학 수업에서 학생들은 스크래치의 기본적인 기능을 배우고 이후 자신만의 게임과 시뮬레이션을 만드는 프로그래밍 수업에 참여하였다. 연구결과, 통제집단은 수학적 역량이 통계적으로 유의하게 향상되지 않았지만, 실험집단은 수학적 역량의 발달이 나타났고, 특히 절차와 알고리즘의 비교 및 실행능력에서 가장 큰 향상을 보였다.

Falloon(2016)은 32명의 초등학교 1-2학년 학생을 대상으로 도형 학습에서 스크래치 주니어(ScratchJr)를 사용하여 간단한 블록 기반 코딩을 할 때 아동이 사용하는 사고 유형을 조사하였다. 스크래치를 사용하기 전 학생들은 정사각형, 직사각형, 정삼각형과 같은 기본적인 도형을 칠판에 그리고 이들의 수학적 속성에 대해 토론하는 수업에 참여하였다. 이후 두 명의 학생이 한 조를 이루어 스크래치를 사용하여 정사각형, 직사각형, 정삼각형을 만드는 활동을 하였다. 학생들의 스크래치 활동과 조별 토론 내용은 Brennan, Resnick(2012)의 CT 이론적 틀과 Bloom의 분류학을 통해 분석되었다. 분석결과, 스크래치를 사용한 도형구성 활동은 학생이 높은 수준의 사고를 발휘할 수 있는 효과적인 도구가 될 수 있음을 보였다. 비록 학생 활동의 절반이 Bloom의 분류학에서 상대적으로 낮은 수준의 사고인 '기억하기'와 '이해하기'에 집중되었지만, 보다 높은 수준의 사고방식인 '분석하기'와 '평가하기'를 하는 모습 역시 나타났다. 학생들은 스크래치 코드를 디버깅할 때 주로 분석적 사고를 사용하였으며, 이는 동료와 협업을 통해 이루어지는 경향이 있었다. 특히 학생들은 코드를 무작위로 만들고 실행 결과를 확인하는 시행착오적인 방법보다 코드를 실행했을 때 예상되는 결과를 동료와 토론하고 코드를 만드는 방식으로 예측에 기반한 분석적 사고를 하는 모습을 볼 수 있었다. 이러한 결과는 연구 참가자가 초등학교 저학년이었다는 점에서 고무적인 결과이며, 수학 수업에 컴퓨터 프로그래밍(코딩)을 사용하는 것이 학생이 사고 능력과 역량 강화에 긍정적인 영향을 줄 수 있음을 시사했다.

분석된 논문에서 스크래치는 대부분 초등학교 고학년과 중학교 학생들의 CT 통합 수학교육에 사용되었다. Park 외(2018)는 스크래치나 엔트리(Entry)와 같은 블록기반 프로그래밍(코딩) 도구가 유아와 초등학교 저학년 학생들이 사용하기에는 어렵다는 것을 지적하면서, 초등학교 1~2학년을 대상으로 손으로 조작할 수 있는 블록놀이에 스마트 기능을 더한 모블로(Moblo)기반 학습을 실시하였다. 이러한 텐저블(tangible) 코딩교육은 초등학교 저학년 학생들의 CT 발달과 협력적 문제해결에 긍정적인 영향을 주었고, 수학에 대한 태도와 코딩교육에 대한 인식에서도 효과적이었다.

선행연구에서는 시각적 프로그래밍(코딩) 도구뿐만 아니라 언어적 프로그래밍(코딩) 도구가 중학교 학생의 CT 및 수학 학습에 효과적임을 시사했다. 예를 들어, Lee, Choi-Koh(2018)는 교사의 시연과 학습자의 모방·창작으로 구성된 시연중심모델에 예비교사들의 멘토링 활동을 결합한 교수학습 모형을 설계하였다. 총 7차시로 구성된 수업에서 중학교 1학년 학생들은 파이썬 프로그램에 대한 기초 학습을 한 뒤, 이를 활용하여 좌표평면과 일차함

[Table 4] Research studies that used programming(coding) as an instructional intervention

Author (Year)	Participants	Intervention	Duration	Method	Results
Taylor et al. (2010)	9-10 year-old (n=60)	Scratch-based game programming(coding) using an interactive whiteboard	One semester	Qual	A learning environment for problem solving and collaboration
Ke(2014)	Middle school (n=64)	Scratch-based mathematical game programming(coding)	12 lessons	Mixed	Development of positive dispositions toward math and reflection on math experience
Calao et al. (2015)	6th grade (n=42)	Scratch-based game and simulation-based programming(coding)	12 weeks	Quan	Improvement in mathematical processes of modeling, reasoning, and problem solving
Park, Kang (2015)	6th grade (n=205)	MIT's Creative Computing-based Scratch	12 lessons	Quan	Influence of logical, mathematical and problem solving abilities on learning flow as well as indirectly on logic, creativity, and CT
Falloon (2016)	1st-2nd grades (n=32)	Creating basic shapes using Scratch(Jnr.)	5 lessons	Qual	An effective means of exercising general and higher order thinking
Jun, Yoon (2016)	8th grade (n=1)	Cow farm modeling, Sierpinski triangle using probability and Omok game with NetLogo	4 semesters	Qual	Positive influence on logical and creative thinking by spontaneously implementing mathematical ideas
Gadanidis et al. (2017)	Elementary PTs (n=143)	Integration of programming(coding) and CT into math	18 lessons	Qual	Development of conceptual understanding of math concepts and positive attitude toward CT in math education
Han (2017)	5th grade (n=47)	Play-based Scratch coding in math	20 lessons	Quan	Development of CT concept and CT practice
Song (2017)	9th grade (n=56)	Scratch-based game programming(coding): trigonometric functions	4 lessons	Quan	Improvement in perceived value of math
Dejarnette (2018)	11th-12th grades (n=23)	Learning trigonometric functions using a visual programming(coding) environment(Etoys)	2 days	Qual	Development of understanding of sine and cosine functions
Gadanidis et al. (2018)	3rd-6th grades (n=415)	Learning group theory through Scratch, Google's Blockly, and hands-on activities	One year[6)]	Qual	A learning environment for engaging young children with ideas of group theory
Park et al. (2018)	1st-2nd grades (n=40)	Game-based learning using a tangible coding block(Moblo)	15 lessons	Mixed	Developments of CT, collaborative problem solving ability, and attitudes toward math and coding
Kim et al. (2018)	7th grade (n=41)	Game-based coding for pattern generalization activities	16 lessons	Mixed	Positive influence on critical thinking, mathematization, abstraction, and automation, as well as improvement in motivation and interests
Lee, Choi-Koh (2018)	7th grade (n=50); Secondary PTs(n=25)	DM^3(Demonstration, Modeling, Making, and Mentoring) program using Python coding	7 lessons	Quan	Positive influence on students' and PTs' self-efficacy in coding as well as on PTs' perspectives on coding in education
Lee, Jung (2019)	8th grade (n=20)	Learning statistics using Python coding	28 lessons	Quan	Development of problem solving ability and increased-interest in both math and coding

* PT indicates pre-service teachers.
* Qual and Quan indicate qualitative and quantitative research methods, respectively.

수에 대해 학습하였다. 연구결과, 파이썬을 활용한 수학교육은 프로그래밍(코딩)에 대한 학생의 자기효능감뿐만 아니라 보조교사로 참여한 예비교사의 자기효능감에도 긍정적인 변화를 주었다. 또한, 수업 후 코딩교육에 대한 예비교사들의 인식에도 긍정적인 변화를 볼 수 있었다. 이 연구는 CT 통합 수학교육에서 상대적으로 미흡하게 다루어지고 있는 (예비)교사의 자기효능감과 코딩수업에 대한 인식을 포함했다는 점에서 연구가 시사하는 바가 크다.

예비교사를 대상으로 수행된 다른 연구로는 Gadanidis, Cendros, Floyd, Namukasa(2017)의 연구가 있었다. Gadanidis 외는 143명의 초등학교 예비교사를 대상으로 수학에 대한 개념적 이해 향상과 CT를 통합한 수학 수업 설계를 돕기 위해 18차시로 이루어진 수업을 진행하였다. 예비교사들은 알고리즘과 코딩에 대한 교육을 시작으로 다양한 수학적 내용 영역(기하, 확률, 패턴 분석, 대수, 측정, 수 개념)에 CT를 통합한 수업에 참여하였다. 수업 후 예비교사들의 온라인 토론 내용과 수업에 관한 자기보고서를 분석한 결과, 예비교사들은 코딩을 통해 컴퓨터와 의사소통하는 과정을 수학적 과정으로 생각하였고, 프로그래밍(코딩)을 수학교육에서 사용할 수 있음을 시사했다. 예컨대, 예비교사는 프로그래밍(코딩)을 배우는 학생들이 컴퓨터 프로그래밍(코딩)을 배우는 것뿐만 아니라 수학적 언어의 힘을 배우는 것이라고 인식했다. 또한, 예비교사들은 CT를 다른 교과(예를 들어, 예술과 음악)와 통합 가능한 사고로 생각하였고, 창의성과 논리적 사고와도 연결시켰다. 전체적으로 CT 기반 수학 수업에 참여한 예비교사들의 프로그래밍(코딩)과 수학 교수에 관한 태도가 긍정적으로 변화였고, CT 활동이 교사 자신들의 수학적 개념 이해를 발달시켰다고 진술하였다.

(2) 로봇을 활용한 연구(Robotics)

로봇은 학생의 흥미를 높여줄 뿐만 아니라 로봇의 움직임을 조종하기 위해 프로그래밍(코딩) 언어 사용이 필요하기 때문에 CT 향상을 위한 도구로 자주 활용된다(Shute et al., 2017). 본 연구에서 검토된 학생의 인지적·정의적 영역 발달을 위해 로봇을 활용한 국내논문은 5편이 있었지만 국외논문은 한 편도 찾을 수 없었다([Table 5] 참고). 국내에서 수행된 로봇을 활용한 수학교육 연구를 살펴보면, 연구 참여자는 유아(Kim & Kim 2019; Lee & Sung, 2017)부터 초등학생(Kim, 2013; Park & Kim, 2010)과 고등학생(Rim, Choi, & Noh, 2014)까지 다양하였고, 사용된 로봇 역시 다양했다.

먼저, Lee, Sung(2017)은 만 4세 유아의 수학적 문제해결력 증진을 위해 코딩용 로봇(비봇)을 활용한 13차시 수업을 구성하였다. 연구에 참여한 30명의 유아는 실험집단(15명)과 통제집단(15명)으로 분류되었다. 실험집단에 속한 유아는 다양한 실생활 맥락에서 로봇을 원하는 위치로 이동시키면서 문제를 해결해 나가는 활동을 하였고,

[Table 5] Research studies that used Robotics as an instructional intervention

Author (Year)	Participants	Intervention	Duration	Method	Results
Park, Kim (2010)	5th grade (n=56)	Robot(Pro-Bot)-integrated group learning in math	16 lessons	Quan	Development of mathematical problem solving ability and learning attitude
Kim (2013)	4th grade (n=31)	Robot(Roamer)-integrated group learning in math	10 lessons	Mixed	Positive influence on interest in math but not on mathematical performance
Rim et al. (2014)	High school (n=11)	Robot(Mindstorm)-based programming(coding)	10 lessons	Mixed	Positive influence on logical and critical thinking
Lee, Sung (2017)	4-year-old (n=30)	Mathematical problem solving through coding robot(Bee-Bot)	13 lessons	Quan	Development of problem solving ability (algebra, measurement, and statistics)
Kim, Kim (2019)	5-year-old (n=38)	Robot-based education	23 lessons	Quan	Development of creative problem solving ability and logic-mathematical knowledge

* Qual and Quan indicate qualitative and quantitative research methods, respectively.

6) 이 연구는 1년 프로젝트의 하나의 부분으로 시행되었다.

교사는 유아가 로봇을 활용하여 주어진 문제를 스스로 해결하도록 도와주는 조력자 역할을 하였다. 통제집단에 속한 유아들은 담임교사를 통해 비슷한 생활 주제에 따른 누리과정 수업을 받았다. 연구결과, 수와 연산, 대수, 기하, 측정, 통계로 이루어진 수학적 문제해결력에서 실험집단이 통제집단보다 유의하게 높은 점수를 받았다. 세부적으로 로봇을 통한 수학교육은 유아의 대수, 측정, 통계 영역에 긍정적인 영향을 미치는 것으로 나타났다.

Park, Kim(2010)은 LOGO 프로그래밍(코딩) 언어에 따라 움직이는 교육용 로봇 프로봇(Pro-Bot)을 활용하여 16차시 초등학교 5학년 수학 교육과정을 구성하였고, 유사하게 Kim(2013)은 교육용 로봇 로머(Roamer)을 활용한 10차시 4학년 수학 교육과정을 설계하였다. 두 연구에서 로봇을 활용한 수학 학습은 모두 모둠별로 진행되었다. Park, Kim은 이러한 실험환경을 전통적인 수학 수업을 받은 통제집단과 비교하였고, 분석 결과 로봇을 활용한 초등학교 수학교육은 학생의 수학적 문제해결과 학습 태도에 긍정적인 영향을 주는 것으로 나타났다. 반면 Kim은 사전-사후 검사를 통해 학생의 수학성취도와 흥미에서의 차이를 조사하였는데, 성취에서는 유의한 차이가 나타나지 않았지만, 수학 수업에서 로봇 활용에 대한 학생의 흥미와 만족도 대체로 높게 나타났다. 로봇의 활용은 공식과 문제 풀이 위주의 전통적인 수학교육에서 벗어나 학생의 흥미와 동기를 북돋을 수 있다는 점에서 긍정적인 효과를 줄 수 있지만, 후속연구에서는 로봇 활용이 학생의 놀이를 위한 도구로만 그치지 않도록 CT와 수학적 역량 향상을 위한 교육과정 구성이 요구된다.

(3) 프로그래밍(코딩) 아닌 컴퓨팅 도구를 사용한 연구

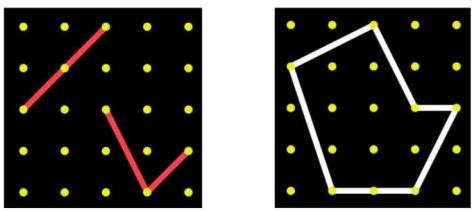
CT가 프로그래밍(코딩)에 국한되는 것이 아니지만(Wing, 2006; 2017), 선행연구에서는 컴퓨터 프로그래밍(코딩) 도구 또는 로봇 프로그래밍(코딩)을 통해 CT와 수학적 이해의 발달을 연구하는 경향이 있었다. 이에 반해, Berkaliev 외(2014)와 Pei, Weintrop, Wilensky(2018) 등은 프로그래밍(코딩)이 아닌 컴퓨팅 상황에서 CT 사용과 수학적 개념이해를 조사하였다([Table 6] 참고). 예를 들어, Pei 외는 고등학교 수학에서 CT 활동과 수학적 사고 습관(mathematical habits of mind)의 발달을 위해 설계된 수학 마이크로월드인 Lattice Land를 소개했다. Lattice Land에서 14명의 고등학생은 평면 위에 동일한 간격으로 찍힌 격자점(lattice point)을 통해 다각형을 그리고 이를 수정하는 과정에서 다각형의 넓이 개념을 탐구하였다([Fig. 2] 참고). 활동 초반에 학생들은 Lattice Land에 그려지는 비(非)전형적인 다각형의 넓이를 구하는 문제에서 이를 삼각형 또는 사각형과 같은 넓이 계산이 가능한 전형적인 도형으로 분해하려 하지 않았다. 하지만 Lattice Land에서 모둠별 토의를 진행하면서 학생들은 주어진 다각형을 작은 삼각형과 사각형으로 분할하여 넓이를 더하는 방법 또는 다각형을 사각형들의 합으로 만든 후 넓이의 합을 반으로 나누는 방법 등을 사용하여 다각형의 넓이를 구하는 모습을 보였고, 모든 학생이 각기 다른 방식으로 다각형을 분해하여 넓이를 구하였다. 이러한 과정에서 학생들은 Weintrop 외(2016)가 기술한 CT 요소인 모델링과 시뮬레이션을 사용하였고, 무엇보다 CT의 핵심요소 중 하나인 문제를 작은 단위로 분해하는 활동을 하였다. 또한, 수학적 사고 습관의 중요한 요소인

[Table 6] Research studies that used non-programming(coding) computing tools as an instructional intervention

Author (Year)	Participants	Intervention	Duration	Method	Results
Berkaliev et al.(2013)	Undergrads (n=70)	Using CT and CT tools in solving mathematical problems	One semester	Quan	Frequent use of CT tools and its positive influence on problem solving ability
Park et al. (2014)	5th grade (n=55)	Freudenthal's RME-based algorithm education	12 lessons	Quan	Development of problem solving ability
Pei et al. (2018)	11th grade (n=14)	CT-integrated math microworld (Lattice Land)	6 days	Qual	A learning environment for cultivating CT practices and mathematical habits of mind

* Qual and Quan indicate qualitative and quantitative research methods, respectively.

시각화, 예측, 실험, 일반화 등을 반복적으로 사용하는 모습을 보였다. 나아가 학생들은 다각형 내의 있는 격자점의 개수와 다각형 위에 있는 격자점의 개수로 다각형의 넓이는 구하는 일반적인 공식(Pick's theorem)을 도출하는 과정에서 CT의 핵심요소인 추상화뿐만 아니라 패턴인식과 엄밀한 용어사용과 같은 수학적으로 사고하는 모습을 보였다. Pei 외의 연구는 CT의 발달이 반드시 프로그래밍(코딩)을 통해 이루어지는 사고가 아니고, "컴퓨팅이 문제해결과 표현도구가 될 때 필요한 사고"(Chang, 2017, p. 560)임을 실증 연구를 통해 보였다는 데 있어 연구의 의의가 있다.



[Fig. 2] Lattice Land used in Pei et al.'s(2018) study

(4) 컴퓨팅 도구를 사용하지 않은 연구

Sung, Ahn, Black(2017)은 CT가 컴퓨터 프로그래밍(코딩)뿐만 아니고 문제해결에서 컴퓨터 과학자가 하는 사고방식과 관련되어있음에도 불구하고, 대부분의 연구는 공학적 도구를 사용한 교수학적 설계를 통해 CT의 발달이나 CT 활동을 통한 교과 관련 개념이해에 집중되었다고 지적하였다. 본 동향연구에서도 컴퓨팅 도구를 사용하지 않았던 연구가 두 편에 불과했다는 사실이 Sung 외의 주장을 지지한다([Table 7] 참고). Sung 외는 초등학교 저학년에 프로그래밍(코딩)을 도입하기 전 학생들의 CT 향상에 의미 있는 활동에 관해 연구하였다. 구체적으로, 66명의 K-1학년 학생들을 대상으로 수학적 문제해결에서 체화된(embodied) 활동과 계산적 관점(computational perspective)[7]을 가지는 것이 수학 개념의 이해와 CT 발달에 미치는 영향을 보고자하였다. 이를 위해, 66명의 학생을 체화된 활동의 정도와 계산적 관점의 유무에 따라 4개의 그룹으로 나누고 2×2 요인실험분석(factorial experiment)을 실시하였다. 체화된 활동의 정도는 몸 전체를 사용하여 크기가 큰 대상을 조작하는 활동(높은 수준)과 손을 사용하여 종이와 같은 작은 대상을 조작하는 활동(낮은 수준)으로 나누어졌다. 체화된 활동에서 계산적 관점의 유무(有無)는 학습자가 동료의 움직임을 지시하고 조종하여 문제를 해결할 수 있는지(有), 자신의 몸을 움직여 문제를 해결하는지(無)에 따라 구분하였다. 동료의 움직임을 통제하기 위해서는 명확한 설명을 통해 자신의 지식을 절차적 단계에 따라 표현해야하므로(프로그래밍에서 명령문과 같은 역할) 이는 컴퓨터 과학자의 사고과정과 유사한 역할을 한다고 보았다. 4개의 집단은 수직선 만들기, 덧셈, 뺄셈 맥락에서 서로 다른 활동을 실시하였고, 이후 각 활동은 스크래치(주니어)를 통해 프로그래밍(코딩) 활동을 실시하였다. 연구결과, 계산적 관점으로 체화된 활동을 수행한 실험집단은 수학적 이해와 프로그래밍(코딩) 능력이 향상되었으며, 계산적 관점을 가지는 것은 체화된 활동의 유무와 관계없이 학생의 프로그래밍(코딩) 개념이해를 도왔다. 이 연구는 초등학교 저학년에서 프로그래밍(코딩) 수업을 도입하기 전 계산적 관점으로 체화된 활동을 제공하는 것이 학생의 프로그래밍(코딩)과 해당 교과 내용 이해에 효과적임을 보였다.

Costa, Campos, Guerrero(2017)는 수학 교수에서 CT의 구성요소를 통합한 문제구성이 학생의 수학적 문제해결력에 미치는 영향에 대해 조사하였다. 이를 위해, 46명의 8학년 학생을 실험집단(23명)과 통제집단(23명)으로 나누고 통제집단에게는 전통적인 확률통계 문항을 제공하였고, 실험집단에게는 Barr, Stephenson(2011)이 제안한 CT의 특성(자료수집, 분석, 표현, 문제분해, 추상화, 알고리즘, 자동화, 병렬화, 시뮬레이션)에 맞게 재구성된 확률통계 문항을 제공하였다. 두 집단은 같은 교사로부터 60분간 해당 문항을 통한 수학 학습을 실시하였다. 이후 교사는 일주일간 두 집단에게 해당되는 문항을 계속해서 과제로 제시하였다. 학생의 문제해결력 평가는 PISA 2012에서 제공한 확률통계 문항을 임의 추출하여 사용하였다. 두 집단의 정답률을 비교한 결과, 문제해결력에 관한 실험집단 학생들의 평균이 통제집단보다 높았다. 또한, 실험집단은 통제집단과 비교하여 문제풀이에 10분정도 더 많은 시간을 사용한 것으로 나타났다. 비모수통계를

7) Berland, Wilensky(2015)의 정의에 따라 계산적 관점은 문제해결에서 컴퓨터 과학자나 프로그래머가 하는 논리적, 분석적, 구성적 사고방식을 강조하는 것으로 학습자가 컴퓨터 과학이 아닌 다양한 분야에서 CT를 사용하여 문제해결을 하는 것을 의미한다.

[Table 7] Research studies that used non-computing tools as an instructional intervention

Author (Year)	Participants	Intervention	Duration	Method	Results
Sung et al. (2017)	K-1st grades (n=66)	Practicing computational perspectives through embodied activities	2 weeks	Quan	Development of mathematical understanding and programming(coding) skill
Costa et al. (2017)	8th grade (n=46)	Practical activities using mathematics problems in conformity with CT	8 days	Quan	Development of problem solving skill as compared to using traditional math problems

* Qual and Quan indicate qualitative and quantitative research methods, respectively.

사용하여 두 집단간 정답률의 차이를 분석한 결과 실험집단의 문제해결력이 통제집단보다 유의하게 높았다. 이러한 결과는 CT 역량과 관련된 수학 문항 구성은 학생의 수학 문제해결력에 긍정적인 영향을 줄 수 있음을 시사했다. 추가적으로 학생에게 실시된 설문 답안을 분석한 결과 실험집단은 통제집단에 비해 과제를 해결하기 위해 동료나 교사의 도움이 더 많이 필요로 했지만, 수업과 과제가 문제해결에 도움이 되었고 실제 PISA 문항이 덜 어렵게 느껴졌다고 답했다. 하지만, Costa 외의 연구는 집단을 분류할 때 학생의 나이와 성별의 동질성은 검토했지만 집단 간 문제해결력이 사전에 유의한 차이가 없는지는 검토하지 않았다는 점에서 연구의 한계를 가진다.

종합하면, 교수학적 처치 방안에 관한 선행연구는 CT를 통합한 수학교육이 CT의 발달과 수학개념에 대한 이해뿐만 아니라 문제해결력과 논리·비판적 사고와 같은 사고역량의 발달에 효과적임을 보였다. 또한, 이러한 교육이 프로그래밍(코딩)과 수학교과에 대한 정의적 영역 발달에 긍정적인 영향을 주었다. 하지만 교수학적 처치방안에 관한 연구(특히 국내연구)는 대부분 로봇 프로그래밍을 포함한 프로그래밍(코딩) 교육에 집중되고 있었다. 이러한 결과는 CT가 컴퓨터 과학자가 하는 사고방식이며 특히 자동화과정이 주로 프로그래밍(코딩)을 통해 이루어진다는 점에서 자연스러운 결과라고 생각할 수 있지만, 자칫 CT를 프로그래밍(코딩)교육과 동일시하는 오류를 낳게 할 수 있다. Wing은 CT 교육이 프로그래밍을 의미하는 것이 아니며 컴퓨터가 반드시 필요한 것이 아니라고 주장했다(Wing, 2008; 2017). 이는 CT의 핵심이 추상화이고 추상화는 문제해결에서 핵심적인 요소를 파악하는 자동화 이전의 사고과정이기 때문이다. CT 개념에 대한 의미를 재고하여 CT와 수학교과의 통합이 보다 의미 있게 진행되기 위해서는 프로그래밍(코딩)교육 뿐만 아니라 다양한 교수학적 설계를 통한 두 학문 간의 통합이 요구된다.

2. 논의

1) 계산적 사고와 수학적 사고

제 4차 산업혁명 시대의 도래로 CT 역량의 중요성이 강조됨에 따라 수학교육에서 CT에 대한 규명이 전제되어야 한다는 당위성에 의해 CT와 수학적 사고 간의 관계에 주목할 필요가 있다. 하지만, 선행연구에서는 CT와 수학적 사고와의 관계를 핵심 주제로 다루었던 이론적 또는 실증적 연구가 이루어지지 않았고, 이러한 관계는 CT 연구에서 작은 장(章) 또는 단락에서 제한적으로 기술되었다. 이는 CT에 대한 정의가 최근 컴퓨터 과학자들을 중심으로 이루어져 왔기 때문일 수 있겠으나 앞서 살펴본 선행연구들에서 수학적 사고를 너무 단순하게 접근하고 있다는 점에서 추상화와 문제해결에 관한 논의가 필요하다.

(1) 추상화

Wing(2008)은 컴퓨터 과학에서 사용하는 추상화가 수학적 추상화보다 복잡하다는 것을 주장하면서 수학적 추상화의 예로 두 정수의 덧셈을 언급하였다. 수학적 추상화의 개념을 정수의 덧셈의 예로 한정해서 설명하는 것은 수학적 추상화가 가지는 복잡성에 대해 충분한 이해에 근거하는지 의문을 제기하게 한다.

여기에 추상화를 바라보는 두 학문 간에 차이가 있다. 두 정수의 덧셈은 피가수와 가수 중에 음의 정수가 있는 경우 수학에서는 간단히 뺄셈개념으로 처리한다. 그러나 컴퓨터 과학에서는 뺄셈도 가수의 보수를 더한 후 상위 자릿수를 제거하고(overflow), 답을 취하는 덧셈의 개념으로 해결한다. 예를 들어 7-3=7+7-10으로 처리하여 4를 얻는다. 이것은 십진법으로 알기 쉽게 정리한 것이고 컴퓨터상에서는 2진수의 수로 분해되고 다시 결합되는 일

련의 복잡한 알고리즘으로 처리되는 것이다. 이런 과정을 컴퓨터 과학자들은 CT의 추상화가 훨씬 복잡하다고 생각한 것이다. 그러나 수학자나 수학교육자는 이 일련의 알고리즘이 추상화이긴 하나 수학에서는 주어진 문제의 해법을 찾기 위해 분석하고 종합하는 과정을 반복하거나 수학화 과정을 통해, 아니면 문제해결의 전략을 통해 식을 세우고 해법을 찾아가는 매우 고차원적 사고과정을 추상화에 포함하고 있다. 반면 컴퓨터 과학에서는 이 복잡한 알고리즘이 컴퓨팅 기기의 도움으로 처리되기 때문에 문제해결자는 디버깅 과정을 거치면서 잘 구성하면 되는 것이고, 수학교육자는 이를 알고리즘에 의한 계산적 문제해결로 보는 것이다. 이 부분이 CT와 수학적 사고와 연결고리인 셈이나 수학적 사고는 훨씬 다양하고 복잡하다.

또한, Sneider 외(2014)가 벤다이어그램으로 시각화한 CT와 수학적 사고와의 관계는 CT와 수학적 사고를 인간의 머릿속에서 이루어지는 사고과정의 관점에서 바라보지 않았고, 사고과정, 교과영역, 교과활동 등을 혼합하여 기술함으로써 논점을 흐리는 경향이 있다. 또한 Barr 외(2011)의 논문에서 ISTE와 CSTA에서 주간한 세미나를 통해 CT와 수학적 사고와의 차이를 3가지로 설명했지만, 이는 사고 간의 차이를 규명하였다기보다 CT의 특성을 기술하는 정도로 다루어졌다는 점에서 연구의 한계를 가진다. 그나마 앞으로 후속연구에 크게 기여할만한 Falloon(2016)의 수학·과학영역을 다룬 연구에서는 분석틀로 Bloom의 분류를 사용하였다. 두 교과영역의 융합적 접근을 위해 초등학교 저학년 학생을 대상으로 연구를 수행하기에 교육적으로 좀 더 일반적인 Bloom의 분류를 따른 것으로 이해할 수는 있으나 수학적 사고는 2015 교육과정에서 언급된 6가지 역량[8] 외에도 다양하다. 즉, Bloom의 후기분류인 기억하기, 이해하기, 적용하기, 분석하기, 평가하기, 창조하기에는 CT에서도 일부 관여돼있는 공통적인 사고로써 수학의 논리성을 기반으로 한 추론적 추상화를 분석하기 이외는 포함하고 있지 않다는 점이다. 수학의 추론적 사고에는 분석과 종합, 특수화와 일반화, 귀납과 연역 등이 해당되는데 Bloom의 '분석하기'보다 상위 수준에 해당되는 고차원적 사고들(high-oder thinking)이며 최근에는 창의와 융합적 사고를 강화하고 있어 이 또한 Bloom의 가장 높은 수준의 사고인 것이다.

(2) 문제해결

여러 학자들은 수학적 사고와 CT 간의 공통적인 관점으로 문제해결을 언급하였다. diSessa(2018)는 CT가 문제해결을 목적으로 한다는 Wing(2006)의 주장에 따라 CT의 사고과정이 Pólya의 문제해결과정과 유사함을 설명하였다. 이보다 앞서 Weintrop 외(2016)는 계산적 문제해결이라는 큰 주제 아래 하위요소들로 7가지 하위요소를 포함하였다([Table 2] 참고). 즉, 계산적 문제해결에서 주로 쓰는 반복, 추측, 시행착오와 같은 문제해결 전략(Barr et al., 2011)은 우리가 일상생활에서나 교과시간에 어떤 기기(예: 컴퓨터나 휴대폰과 같은 대용품)가 주어졌을 때 일반화를 목적으로 누구나 쉽게 시도해보는 전략이다. 그러나 수학교육에서 비정형적 요소의 문제해결은 수학적 추상화와 마찬가지로 상위수준 기술(Stanic & Kilpatrick, 1989, p. 15)로써 유연성, 정교성, 유창성과 같은 창의적 요소마저 포함되어 있다. 다시 말해, CT는 컴퓨팅 도구를 통한 자동화로 반복, 추측, 시행착오라는 다소 단순한 문제해결 전략에 의존하면서 일반화를 위한 문제의 공식화를 통해 현대에 우리가 일상생활에서 부딪히는 문제들에 즉각적인 해법을 제공하는 장점을 가지고 있다. 그러므로 이 계산적 문제해결력은 모든 교과에 적용가능하며 그 교과의 특성과 아울러 새로운 통찰로 이어질 수 있다는 점에서 앞으로 CT가 갖는 잠재력인 것이다.

후속연구에서는 CT와 수학적 사고의 특성과 이들 간의 관계를 보다 명확히 조명하기 위해 수학교육자와 컴퓨터교육학자 간의 심도 있는 논의와 분석이 요구된다. 학생의 수학적 사고기제 안에 CT가 명확히 규명되는 연구결과를 산출하기 위해서는 서로의 다른 관점을 조절하면서 각 특성을 파악해볼 수 있는 연구 설계에서의 차별성 또는 정교성이 필요해 보인다. 수학적 사고과정과 관련하여 CT의 하위요소 간에 그 관계성을 면밀히 살펴볼 필요가 있으며 그러기 위해서는 마이크로적 접근이 가능한 정성연구의 필요성이 제기되는 부분이기도 하다. 이런 사고과정을 규명한 정성연구를 바탕으로 연구목적의 효과를 정량적으로 조사하는 것도 차후에 단계적으로 요구된다고 하겠다.

---

[8] 문제해결, 추론, 의사소통, 창의·융합, 정보처리능력, 수학적 실천 및 태도 역량이다.

2) 수학교육에서 SW 교육

최근 학교현장의 큰 변화로는 2018년부터 초·중등학교에서 SW 교육을 의무화하여 정보교과를 필수 교과로 법제화한 것이다(Ministry of Education, 2015b). 여기에 정보교과의 코딩[9]에 관한 교육을 의무적으로 시행하도록 포함하였다(Ministry of Education, 2016). 그동안 우리나라 수학교육에서는 제 6차 교육과정(Ministry of Education, 1992)에서 공학도구 활용을 명시한 이후 그래핑 계산기 활용뿐만 아니라 LOGO, Excel, GSP(Geometer's Sketch Pad), Cabri, GraEq 등 다양한 소프트웨어를 활용한 컴퓨터 환경에서의 SW 교육은 CT의 중요성이 부각된 후에도 지속적으로 이루어져 왔다(예: Choi-Koh et al., 2015; Kim, 2015; Son, 2011). 다만 대부분의 연구가 수학적 사고 내에서 이루어져 왔다는 점이다. CT가 강조되기 이전 수학교육에서 SW 교육을 살펴보면, Choi-Koh(2003)은 수학적 사고 중 귀납적 사고, 연역적 사고, 그리고 창의적 사고와의 관계를 공학도구활용 환경에서 조사하였는데 창의적 사고는 그 특성상 문제해결력의 산물로써 특히 수렴적 사고를 요구하는 연역적 사고보다는 발산적 사고를 요구하는 귀납적 사고에서의 다양성과 크게 관련되어 있음을 주장하였고, 두 집단의 학생들의 창의적 사고과정이 용이하지 않음을 밝히면서 교수학적 배려가 중요한 변수임을 시사하였다. 또한 Choi-Koh(2005)는 기술공학의 시대적 발달을 '거대한 물결'로 묘사하며 프로그래밍(코딩) 도구와 문제해결 도구로 구별하여 세대적 발달단계로 분류하면서 앞으로 정보통신기술(ICT) 시대를 대비하여 문제해결로써 도구적 활용에 따른 변화가 다가올 시대를 주도할 것임을 예측하였다. 이는 문제해결적 관점으로 CT를 보려는 Berland, Wilensky(2015) 또는 Perez(2018)과 일치한다.

수학교과와 CT를 통합할 때 사용하는 프로그래밍 언어는 가능한 대로 코딩 방식이 간편해야 하며 사용 방법을 익히는 것이 부담스럽지 않아야 한다(Chang, 2017). Excel이나 GSP와 같은 SW가 수학의 문제해결을 위한 도구로써 주는 장점은 지필환경에서 접근하기 어려웠던 재귀적이며 자동화가 가능한 알고리즘의 능력으로 즉각적인 해법을 제공할 수 있는 것뿐만 아니라 프로그래밍 언어로 소통할 필요가 없이 프로그래밍 빌트 인(built in) 소프트웨어들이란 점이며 이런 획기적인 변화는 일상 언어로 표현이 가능한 LOGO로부터 출발한다(Choi-Koh, 2018, p. 37; Lew & Shin, 1998). 즉, 수학적 언어로 자동변환해주기 때문에 사용자는 수학적 언어로 소통이 가능하여 사용자의 목적인 난해한 수학적 문제해결에 집중할 수 있게 된 것이다. 예를 들어, GSP에는 스크립트 창에 사용자가 시행한 작업과정이 그대로 순서적으로 수록되어 있어 새로운 창에서 언제든지 저장된 스크립트의 반복 재생이 가능하며 다음 단계의 복잡한 문제로 나아갈 수 있다. 사용자가 도구에게 요구하는 도구화(Instrumentalization; Trouche, 2004)가 용이하기 때문에 이러한 도구를 사용자 중심(user-oriented) 도구라고 한다. 이런 사용자 중심 도구들에 의해 그간 수학교육에서는 CT 교육과의 통합이 관심받지 못한 것이며 이러한 관점이 지속된다면 수학교육에서 CT의 교육과 직접적으로 관련되어있는 수치해석, 이산수학, 선형대수와 같은 영역 이외에서 그 필요성과 수학교육자가 주도적으로 CT 교육의 활성화를 추구하기는 여전히 미약할 것으로 예측된다.

또한, 각 교과에서 CT 교육의 통합이 제기되고 있는 요즘 SW 교육에 대한 다양한 연구에도 불구하고 학교현장과의 괴리는 여전히 존재한다. 정보교과에는 SW 교육을 하도록 의무화하고 있으나 현재 학교는 WiFi 제한지역으로 지정하고 있어서 타 교과에서는 자유로운 기기 활용이 제한되어 있을 뿐만 아니라 모든 교육의 성공의 열쇠를 쥐고 있는 교사를 양성하는 대학의 교사교육 역시 시대적 변화에 따라가고 있지 못하다(Jang, 2017; Kang, Lee, & Choi-Koh, 2017; Lee, & Choi-Koh, 2018). 물론 CT의 교육이 컴퓨터 환경이 아닌 환경에도 가능한 것이나 더 효율적인 CT 교육을 실천하기 위해서는 현장에 과감한 투자와 변화 없이는 불가능하다. 수학교실을 확보하여 인터넷 환경에서 CT를 통합한 다양한 교수학습이 이뤄지도록 지원하는 것도 한 방안이다. 따라서 각 교과에서 CT 교육의 중요성으로 인해 현장의 개선된 환경조성이 앞당겨질 수 있길 기대한다.

9) 코딩교육은 영어로 번역하자면 coding education으로 표기될 단어가 존재하지 않는다는 점을 상기하고자 한다. 교육을 코딩하는 뜻이 될 것이기 때문이다. 적절한 표현은 coding in education이다. 그럼에도 우리나라 표현에 코딩교육이 자연스럽게 쓰이는 것은 우리나라 단어조합이 포괄적이고 광의적 뉘앙스를 담아내는 잠재력 때문이다.

3) 연구 설계의 한계성

국내에서 진행된 실험연구는 편향된 연구방법론의 사용으로 연구의 한계를 지닌다. 국내 연구에서 정성연구 방법을 사용한 논문은 Jun, Yoon(2016)의 사례연구가 유일했고, 대부분의 국내 연구에서는 정량 연구방법 또는 정량적 분석에 기반한 혼합연구가 수행되었다. 정량연구와 정성연구는 강조하는 방향이 서로 다르므로 둘 간의 우열을 논할 필요는 없다. 하지만 그만큼 각각의 장단점도 뚜렷하기 때문에 특정 주제에 대한 편향된 연구방법론의 사용은 불완전한 결과를 생산할 수밖에 없다. 예를 들어, 본 연구에서 다수의 연구자가 사용한 선다형 검사지를 통한 통계분석 방법은 학생의 사고 구조를 정확하여 반영할 수 없다는 것에서 연구의 한계를 지닌다(Beggrow, Ha, Nehm, Pearl, & Boone, 2014). 후속연구에서는 정량연구가 가지는 맥락성 부재와 정성 연구가 가지는 객관성 부재가 상호 보완할 수 있도록 다양하고 균형 있는 연구방법론의 사용이 요구된다.

Ⅴ. 결론 및 제언

1. 결론

전 세계적으로 CT를 학교 교육과정에 통합하려는 움직임이 일고 있고, 수학교과는 이러한 움직임의 핵심이 되고 있다. 본 연구에서는 CT에 대한 개념을 Jeannette Wing의 관점에서 출발하여 선행연구에서 다루어진 이론적 틀을 바탕으로 CT와 MT와의 관계를 규명하였고, 나아가 CT를 수학교과에 통합한 국내외 연구동향을 분석하였다. 선행연구를 종합적으로 분석한 결과를 토대로 다음과 같은 결론을 도출하였다.

첫째, 학자마다 CT에 대한 정의가 상이한 만큼(García-Peñalvo & Mendes, 2018; Lye & Koh, 2014) 검토된 선행연구에서는 CT와 MT의 관계가 피상적으로 기술되어 있었으며 여전히 학자들 사이에서 논쟁을 불러일으키고 있다(Barr et al., 2011; Brennan & Resnick, 2012; Grover & Pea, 2013; Shute et al., 2017). 그럼에도 이들을 요약해보면 CT와 MT가 문제해결을 지향한다는 공통점을 지니면서 두 가지 사고의 핵심이 되는 추상화의 방식에서는 차이가 있었다. 앞선 논의 단원에서 살펴보았듯이, 이 두 공통점과 차이점 내에서도 수학적 사고가 갖는 특성이 훨씬 난해하다. 따라서 CT를 잘 이해하고 수학교육자가 중심이 되어 이를 수학적 사고와 관련짓는 접근이 필요하다.

둘째, 수학교육에서 CT에 관한 실험연구는 주로 프로그래밍(코딩)을 통한 학생의 인지적 능력(수학적 문제해결력과 CT 발달 등)과 정의적 영역(흥미, 가치, 태도, 자신감, 자기효능감 등)에서의 향상에 주목하였다. 수학교과에 CT를 통합하여 학생과 교사의 인지적·정의적 영역에서의 변화를 살펴본 연구에서 다양한 처치 프로그램과 연구방법 개선이 필요하다.

먼저 CT 발달 및 CT 도구를 통한 수학교육에서 반드시 컴퓨팅 도구가 필요한 것이 아님을 인식해야 한다(Costa et al., 2017; Sung et al., 2017; Wing, 2008; 2017). 선행연구에서 프로그래밍(코딩) 도구로는 스크래치와 같은 시각적 프로그래밍(코딩) 도구, 파이썬과 같은 언어적 프로그래밍(코딩) 도구, 로봇 프로그래밍(코딩) 도구 등 다양했으며 실험연구의 주를 이루었다. 이는 CT 교육을 코딩(을 활용한) 교육과 동일시하게 인식하는 계기가 되었다. 반면 CT는 문제해결을 효과적으로 해결하기 위해 문제를 분해하고 공식화하는 사고 과정이므로(Wing, 2006; 2008) 컴퓨팅 도구 없이도 CT의 향상(Sung et al., 2017) 및 CT 통합을 통한 수학적 문제해결력 향상(Costa et al., 2017)을 기대할 수 있다. 따라서, 수학교과에서 CT 통합에 관한 후속연구에서는 프로그래밍(코딩) 도구 이외에 다양한 교수학적 프로그램 설계를 통한 처치방안 개선이 요구된다.

2. 제언

1) 수학교육자의 인식변화 필요성

CT에 대한 정의와 분류가 학자마다 다양하다. Chang(2017)은 CT의 고유의 특성에 반복, 재귀, 디버깅, 자동화를, 수학적 사고의 고유 특성에는 수학적 문제해결과 표현, 수학적 개념과 원리로 분류하였다. 여기서 CT의 고유 특성에 속한다는 반복과 재귀가 교육과정 상에서 학습 양의 축소로 사라졌지만 수 연산, 함수, 수열에 기초를 두고 있기에 수학적 사고와 분리한 점에 의문이 든다. 그뿐만 아니라 추상화와 문제해결 관점에 두 학문 간에 차이점이 존재함을 앞 단원에서 언급하였듯이 수학의 특성상 수학적 사고에는 더 고차원적인 추상화와 문제해결

이 포함되어 있다. 앞으로 수학교육자들이 주도하는 실험연구가 더욱 활발해져서 영역 간의 분류가 구체적으로 확립되길 기대한다. 또한, 지난 10년간 CT의 중요성이 강조되었음에도 그동안 수학교육자들은 수학적 사고의 고유한 영역에만 관심 두어왔기에 수학적 사고를 기반으로 하는 CT에 관한 연구에 소홀해 온 것이 사실이다. 미래 4차 산업혁명시대에 필요한 인재양성을 위해서 뿐만 아니라 두 학문의 발전을 위해 본 연구를 통해 수학교육자들의 CT에 대한 인식이 바뀌길 바라며, 두 학문 간이 협업이 활발히 일어날 수 있도록 국가적 차원에서 교육부가 연구 과제를 통해 적극적으로 지원하여야 한다.

2) 수학 교사교육에 대한 연구 필요성

교사 전문성 개발과 예비교사 교육은 K-12 교육과정에서 CT 통합에 핵심적인 요소이다(Barr & Stephenson, 2011). 이를 위해, 미국과 유럽 등 해외에서는 국가적 차원에서 교사가 학교 교육과정에 CT를 통합하여 운영할 수 있도록 교사연수를 실시하고 있으며, 우리나라에서도 지난 4년간 초등학교 교사와 중등학교 교사를 대상으로 코딩교육 관련 직무연수를 실시하였다(Ministry of Education, 2016). 하지만, 본 연구에서 살펴보았듯이 수학교과 또는 수학적 맥락에서 교사들의 CT 통합에 관한 실험연구는 거의 수행되지 않았다. 교사를 대상으로 한 SW 교육 연수가 주로 정보 및 컴퓨터 교과 관련 교사를 대상으로 이루어졌다는 사실을 고려하여 이를 전체 교과로 확장한다 하더라도 교사 대상 연구가 전체의 10% 미만이라는 Lee(2018)의 연구결과는 교사관련 CT 연구의 필요성을 제기한다.

또한, 예비 수학교사에 대한 CT 연수 및 연구 역시 더욱 활발하게 진행되어야 할 필요가 있다. 4차 산업혁명의 시대적 흐름에서 CT는 읽기, 쓰기, 셈하기와 더불어 모든 학생이 갖추어야 할 기본적 소양이고(Wing, 2006), 그 중요성은 시간이 지날수록 더해질 것이다. 하지만, 예비교사들이 CT가 무엇인지에 대한 개념이 부족하고, 어떻게 그들의 수업에서 CT를 통합해야 하는지에 관한 명확한 아이디어가 없는 실정이다(Bower & Falkner, 2015). 이에 Lee, Choi-Koh(2018)은 SW 교육이 현직 수학교사뿐만 아니라 예비 수학교사에게도 활발히 이루어져야 한다고 주장하였다. 나아가, 각 교과에서 CT 통합이 의미 있게 이행되기 위해서는 교사교육 및 연수에서 교사가 CT를 컴퓨터 프로그래밍(코딩)으로 동일시하여 생각하지 않도록 주의를 기울여야 한다. CT 교육을 프로그래밍으로 생각한다면 (예비)수학교사를 위한 CT 교육의 당위성이 설명되지 않는다.

Yadav, Mayfiled, Zhou, Hambrusch, Korb(2014)는 교사들이 CT에 대한 이해를 그들의 교과 맥락 속에서 향상시키는 것이 중요하다고 주장하였다. 그렇지 않으면 교사들은 CT를 추상적으로 이해할 것이고, 교과 관련된 그들의 지식이 활성화되지 않아 CT를 교과수업에 효과적으로 통합하는 것이 어렵기 때문이다. 따라서 교사교육 기간 중 수학교과의 맥락 속에서 CT와 수학적 사고의 통합하는 방법을 모색하는 과정을 경험하게 함으로써 교사의 전문성을 키우게 하여야 한다. 국내 초등학교와 중등학교에서 SW 교육의 의무화가 시행된 지 얼마 지나지 않았다. 교육에서 교사의 역할은 변하고 있지만 그 중심에 교사를 배제할 수 없으므로 CT 통합 교육과정에서 예비교사와 현직교사에 대한 지속적인 교육과 연구가 요구되는 바이다.

3) 다양한 연구방법 필요성

어떤 주제에 대해 특성을 연구하거나 관계성을 규명하고자 할 때 실증적인 자료로 접근할 수 있는 실험연구의 중요성은 아무리 강조해도 지나치지 않는다. 현장의 구성원과 자원을 통한 실증적 자료에 근거하지 않은 이론은 탁상공론에 빠지기 쉽기 때문이다. 이러한 관점에서 현재까지 이뤄진 실험연구들이 (특히 국내에서) 컴퓨터 교육자들 중심으로 이뤄져 왔으며, 국외에서 수행되었던 실험연구는 정성연구와 정량연구가 균형 있게 진행된 반면, 국내연구는 정량연구와 정량적 분석이 주가 되고 정성적 분석은 설문 조사 결과를 인용하는 수준으로 진행된 혼합연구가 주를 이루었다. 혼합 연구방법 동향을 분석한 Kim, Bae, Kim, Lee, Choi(2014)에 따르면, 혼합 연구방법의 유형은 정성적 분석과 정량적 분석의 정도에 따라 다양하다. 하지만, 본 연구에서 분석된 국내연구는 대부분 정량연구와 정량적 분석에 기반한 혼합연구 형태로 편향되게 수행되었다. 따라서 초기에 이론의 기초를 마련하고 관점을 개발하기 위해 맥락적이고 과정중심이며 나아가 토대연구 기법이 수월한 정성적 연구방법의 필요성이 제기된다.

참 고 문 헌

Ananiadou, K., & Claro, M. (2009). 21st century skills and competences for new millennium learners in OECD countries. *OECD Education Working Papers, 41*, OECD Publishing.

Association for Computing Machinery (2014). New report presents recommendations and initiatives to address CS education challenges. Retrieved May 30, 2019 from https://cacm.acm.org/news/172705-new-report-presents-recommendations-and-initiatives-to-address-cs-education-challenges/fulltext

Balanskat, A., & Engelhardt, K. (2015). *Computing our future: Computer programming and coding priorities, school curricula and initiatives across Europe*. Brussels, Belgium: European Schoolnet. Retrieved May 30, 2019 from http://tinyurl.com/zagj3wj

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology, 38*(6), 20 - 23.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *ACM Inroads, 2*(1), 48-54.

Beggrow, E. P., Ha, M., Nehm, R. H., Pearl, D., & Boone, W. J. (2014). Assessing scientific practices using machine-learning methods: How closely do they match clinical interview performance?. *Journal of Science education and Technology, 23*(1), 160-182.

Berkaliev, Z., Devi, S., Fasshauer, G. E., Hickernell, F. J., Kartal, O., Li, X., ... & Zawojewski, J. S. (2014). Initiating a programmatic assessment report. *PRIMUS, 24*(5), 403-420.

Berland, M., & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology, 24*(5), 628 - 647.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining twenty-first century skills. In P. Griffin, B. McGaw, & E. Care (Eds.), *Assessment and teaching of 21st century skills* (pp. 17 - 66). Netherlands: Springer.

Bower, M., & Falkner, K. (2015, January). Computational thinking, the notional machine, pre-service teachers, and research opportunities. *Proceedings of the 17th Australasian Computing Education Conference* (pp. 37-46). Sydney, Australia. Retrieved May 30, 2019 from https://pdfs.semanticscholar.org/c2df/f4fdd833c44015fedff1e9ae480740894a7b.pdf

Brennan, K., & Resnick, M. (2012). Using artifact-based interviews to study the development of computational thinking in interactive media design. *Paper presented at annual American Educational Research Association meeting*. Vancouver, Canada.

Burton, L. (1984). Mathematical thinking: The struggle for meaning. *Journal for Research in Mathematics Education, 15*(1), 35-49.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing mathematical thinking with scratch. In *Design for teaching and learning in a networked world* (pp. 17-27). Springer International Publishing.

Cetin, I., & Dubinsky, E. (2017). Reflective abstraction in computational thinking. *The Journal of Mathematical Behavior, 47*, 70-80.

Chang, K. (2017). A feasibility study on integrating computational thinking into school mathematics. *School Mathematics, 19*(3), 553-570.

Choi-Koh, S. (2003). The effective use of a technology tool for students' mathematical exploration. *Mathematics Education, 42*(5), 647-672.

Choi-Koh, S. (2005). *Push the Excel if you want to do math*. Seoul: Kyungmoonsa.

Choi-Koh, S. (2018). *Problem solving competence. Korean society of mathematical education yearbook 2017: Mathematical Competences and Capabilities in Korea Math Education*(pp. 25-51). Seoul: Kyungmoonsa.

Choi-Koh, S., Ko, H., Gu, N., Kim, N., Kim, R., Kim, H., …, Han, S. (2015). *Journal of Korea society educational studies in mathematics yearbook 2015: Technological tools in mathematics education*. Seoul: Kyungmoonsa.

Computer Science Teachers Association. (2017). *CSTA K-12 Computer Science Standards, Revised 2017*. Computer Science Teachers Association. Retrieved May 30, 2019 from http://www.csteachers.org/standards.

Costa, E. J. F., Campos, L. M. R. S., & Guerrero, D. D. S. (2017, October). Computational thinking in mathematics education: A joint approach to encourage problem-solving ability. *In 2017 IEEE Frontiers in Education Conference* (FIE). IEEE, Indianapolis, IN.

DeJarnette, A. F. (2018). Students' conceptions of sine and cosine functions when representing periodic motion in a visual programming environment. *Journal for Research in Mathematics Education, 49*(4), 390-423.

diSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. Cambridge, MA: MIT Press.

diSessa, A. A. (2018). Computational literacy and “the big picture” concerning computers in mathematics education. *Mathematical Thinking and Learning, 20*(1), 3-31.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning, 32*(6), 576-593.

Gadanidis, G., Cendros, R., Floyd, L., & Namukasa, I. (2017). Computational thinking in mathematics teacher education. *Contemporary Issues in Technology and Teacher Education, 17*(4), 458-477.

Gadanidis, G., Clements, E., & Yiu, C. (2018). Group theory, computational thinking, and young mathematicians. *Mathematical Thinking and Learning, 20*(1), 32-53.

García-Peñalvo, F. J., & Mendes, J. A. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior, 80*, 407-411

Gouws, L., Bradshaw, K., & Wentworth, P. (2013). Computational thinking in educational activities. In J. Carter, I. Utting & A. Clear (Eds.), *The proceedings of the 18th Conference on Innovation and Technology in Computer Science Education* (pp. 10 - 15). Canterbury: ACM.

Grover, S., & Pea, R. (2013). Computational thinking in K - 12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43.

Han, S. (2017). Play-based SW education teaching-learning strategy to improve computational thinking. *Journal of Korea Association of Information Education, 21*(6), 657-664.

Hwang, Z., & Hwang, S. (2017). An analysis of research trends software education for elementary school: Focusing on domestic articles. *Journal of Korea Association of Information Education, 21*(5), 509-525.

Jang, M. (2017). *A study on technological pedagogical content knowledge of middle school mathematics teachers*. Doctoral Dissertation, Chonnam National University.

Jun, Y., & Yoon, J. (2016). Case exploration of a gifted student's spontaneous and creative project activities using NetLogo in a math-information combined class. *The Journal of Science Education for the Gifted, 8*, 145-166.

Kafai, Y., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan, 95*(1), 61 - 65.

Kang, T., Lee, S., & Choi-Koh, S. (2017). Development and implementation of the program for the free learning semester focused on career exploration. *Mathematics Education, 56*(2), 177-191.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education, 73*, 26-39.

Kim, C. (2013). Design and application of math class with robot. *Journal of Korea Association of Information Education, 17*(1), 43-52.

Kim, D., Bae, S., Kim, W., Lee, D., & Choi, S. (2014). Trends of mathematics education research and mixed methods - Focusing on domestic mathematics education journals for the last 10 years. *Communications of Mathematical Education, 23*(3), 303-320.

Kim, H. (2015). *A turtle microworld and computing thinking. Korea society educational studies in mathematics yearbook 2015: Technological tools in mathematics education* (pp. 355-367). Seoul: Kyungmoonsa.

Kim, N., Seo, Y., & Cho, H. (2018). Coding mathematics contents and environment design - Focusing on mathematization and computational thinking. *Journal of Learner-Centered Curriculum and Instruction, 18*, 647-673.

Kim, S. U., & Kim, S. H. (2019). The effect of a robot-based education program on young children's logic-mathematical knowledge and creative problem-solving. *The Journal of Future Early Childhood Education, 26*(1), 209-229.

Lee, Y. (2018). Domestic research trends analysis of software education. *The Journal of Educational Information and Media, 24*(2), 277-301.

Lee, S., & Choi-Koh, S. (2018). The effects of the mathematical program, DM^3 based on coding instruction using Python. *The Journal of Educational Research in Mathematics, 28*(4), 479-499.

Lee, D., & Jung, J. (2019). The effects of middle school mathematical statistics area and Python programming STEAM instruction on problem solving ability and curriculum interest. Journal of the Korea *Academia-Industrial Cooperation Society, 20*(4), 336 - 344.

Lee, Y., & Sung, H. (2017). Influence of program using the coding robot “Bee-Bot” on children's mathematical problem solving ability. *Children's Media Study, 16*(3), 261-281

Lew, H., & Shin, D. (1998). *Mathematics education and*

computer. Seoul: Kyungmoonsa.

Lockwood, E., DeJarnette, A. F., & Thomas, M. (2019). Computing as a mathematical disciplinary practice. *The Journal of Mathematical Behavior*, Advance online publication. https://doi.org/10.1016/j.jmathb.2019.01.004

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior, 41*, 51-61.

Ministry of Education (1992). *The 6th reformed middle school curriculum*. Seoul: The author.

Ministry of Education (2015a). *2015 reformed mathematics curriculum 2015-74* [Supplementary Book 8]. Retrieved July 21, 2019 from http://www.moe.go.kr

Ministry of Education (2015b). *The plans for educating students for the SW-oriented society*. Retrieved July 21, 2019 from http://www.moe.go.kr

Ministry of Education (2016). *The press materials about coding education in the elementary & secondary School*. Retrieved July 21, 2019 from http://www.moe.go.kr

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. Washington, DC: National Academies Press.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Park, J., & Kang, M. (2015). Structural relationships among learners' characters, learning flow, and thinking ability in a Scratch programming course for elementary school students. *The Journal of Elementary Education, 28*(4), 145-170.

Park, J., & Kim, C. (2010). The effects of robot based mathematics learning on learners' attitude and problem solving skills. *The Journal of Korea Association of Computer Education, 13*(5), 71-80.

Park, M., Kim, D., Kim, J., Kim, H., Lee, B., Cho, Y., & Hong, J. (2018). An analysis on the effects of a tangible coding education program. *The Journal of Korea Elementary Education, 29*(4), 23-49.

Park, M., Kim, J., & Kim, T. (2014). The effect of the RME-based algorithmic learning on elementary students' problem solving ability for improving computational thinking. *Korean Journal of Teacher Education, 30*(4), 179-193.

Pei, C., Weintrop, D., & Wilensky, U. (2018). Cultivating computational thinking practices and mathematical habits of mind in Lattice Land. *Mathematical Thinking and Learning, 20*(1), 75-89.

Pérez, A. (2018). A Framework for computational thinking dispositions in mathematics education. *Journal for Research in Mathematics Education, 49*(4), 424-461.

Rim, H., Choi, I., & Noh, S. (2014). A study on the application of robotic programming to promote logical and critical thinking in mathematics education. *Mathematics Education, 53*(3), 413-434.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*(2), 351 - 380.

Shim, K., & Shim, S. (2018). Development of teaching method of mathematics subject with python coding - Focusing on the content of 'prime decomposition' in the middle school mathematics subject of 2015 revised curriculum. *Educational Study, 73*, 43-64.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142-158.

Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Computational thinking in high school science classrooms. *The Science Teacher, 81*(5), 53.

Son, H. (2011). A study on students' conjecturing of geometric properties in dynamic geometry environments using GSP. *School Mathematics, 13*(1), 107-125.

Song, J. (2017). Effects of learning through Scratch-based game programming on students' interest in and perceived value of mathematics curriculum. *Journal of Korea Association of Information Education, 21*(2), 199-208.

Stanic, G. M., & Kilpatrick, J. (1989). Historical perspectives on problem solving in the mathematical curriculum. In R. I. Charles, & E. A. Silver(Eds.) *The Teaching and Assessing of Mathematical Problem Solving*(pp. 1-22). Hillsdalc, NJ: Erlbaum

Sung, W., Ahn, J., & Black, J. B. (2017). Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning, 22*(3), 443-463.

Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia-Social and Behavioral Sciences, 8*, 561-570.

Trouche, L. (2004). Managing the complexity of human/machine interaction in a computerized learning environments: Guiding students' command process

through instrumental orchestrations. *International Journal of Computers for Mathematical Learning, 9*, 281-307.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127-147.

White House (2016). FACT SHEET: President Obama announces computer science for all initiative. Retrieved May. 30, 2019 from https://obamawhitehouse.archives.gov/the-press-office/2016/01/30/fact-sheet-president-obama-announces-computer-science-all-initiative-0

Williams, S. R., & Leatham, K. R. (2017). Journal quality in mathematics education. *Journal for Research in Mathematics Education, 48*(4), 369-396.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366*, 3717-3725.

Wing, J. M. (2014). Computational thinking benefits society. 40th Anniversary Blog of Social Issues in Computing, 2014. Retrieved May. 30, 2019 from http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

Wing, J. M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology, 25*(2), 7-14.

Wing, J. M., & Stanzione, D. (2016). Progress in computational thinking, and expanding the HPC community. *Communications of the ACM, 59*(7), 10-11.

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education, 14*(1), 1-16.