대한수학교육학회지 <학교수학> 제 19 권 제 3 호
Journal of Korea Society Educational Studies in Mathematics
School Mathematics Vol. 19, No. 3, 553 ~ 570. Sep 2017

수학 교과에서 계산적 사고(Computational Thinking)교육[1)]

장 경 윤*

본 연구는 21세기 필수 능력으로 거론되는 계산적 사고의 의미를 살펴보고, 수학교과에서 CT 교육의 가능성 여부와 그 선행 조건을 탐색할 목적으로 수행되었다. 선행연구를 통해 컴퓨터, 학교교육, 수학교육에서의 CT의 정의와 구성 요소를 조사하였으며 본 연구에서는 수학교과에서 CT를 수학적 문제해결 관련 사고로 보았다. CT-컴퓨팅(컴퓨터 활용)-수학교육 세 영역 사이의 관계 고찰에서 컴퓨팅환경에서 유용한 CT이나 수학교육에는 포함되지 않는 영역에 주목하였다. CT와 수학교육의 통합논의에서는 컴퓨터가 전통적 수학교육의 보조 수단으로 허용되는 우리나라 수학교육 현황을 고려할 때, '컴퓨팅 환경에서의 수학적 문제해결'에 주목할 필요가 있다고 보았다. 수학교육에서 CT 교육은 컴퓨팅 환경 조성을 전제로 수학교과에서 수학 관련 과제에 해결을 위한 코딩, 문제해결, STEAM 교육 맥락에서 수학과 CT의 통합을 제시하였으며 이를 위하여 CT 통합을 지원하는 수학교육과정 마련 등 제반 조건을 논의하였다.

I. 서론

최근 기술공학의 사용이 빈번한 사회에서 Computational Thinking은 21세기를 위한 소양으로 거론된다. 미국 NRC는 학생들을 성공하고 번영할 수 있게 준비시키기 위한 국가적인 노력의 일환으로, 차세대 과학규준(Next Generation Science Standards)에 computational thinking을 과학 교과의 핵심 실천에 포함시켰다. 미국 경제의 경쟁력 유지, 타 교과에서의 탐구 지원, 복잡한 문제를 공략하는 역량 강화를 위해, 계산적 사고의 중요성을 과학교과에서 강조한 것이다(NRC, 2011, Yadav, etc., 2016).

우리나라에서도 창의적 문제해결력을 갖춘 미래인재양성에 대한 국가적 관심이 커지면서 computational thinking에 주목하게 되었고 이는 코딩교육의 강화로 나타났다. 우리나라 2015 교육과정은 computational thinking을 '컴퓨팅 사고력'으로 번역하여 컴퓨팅 사고력을 [정보] 교과에서 추구할 역량으로 교육과정에 명시(교육부, 2015b)하였다. 그리고 한국과학창의재단에 소프트웨어 교육을 위한 전담부서가 신설되고 교육부는 2015년 소프트웨어 운영지침을 발표하였으며, 2018년부터 초·중등학교에서 S/W 교육을 의무화하여 [정보] 교과는 필수 교과로 법제화되었다(미래창조과학부·교육부, 2015c). 즉, 우리나라에서 computational thinking 교육은 곧 코딩교육을 의미하는 것으로 보인다.

제 4차 교육과정이후 최근 30여 년간 우리나라 수학교육과정 개정은 지속적인 학습량 감소로 이어져 왔으며, 그 배경에는 수학교육에서 지

* 건국대학교 kchang@konkuk.ac.kr
1) 이 논문은 2014학년도 건국대학교의 연구년교원 지원에 의하여 연구되었음.

식의 습득보다 수학적 소양과 문제해결을 갖춘 인재 양성에 대한 국가적 관심이 있었다. 산업사회와 정보화 사회를 넘어 제 4차 산업혁명에 진입하는 현 시점에서 수학적 소양을 갖춘 융합적 인재에 대한 사회적 요구는 그 어느 때보다 강력하다. 그러므로 학교 수학도 변화하는 사회에 적응하며 수학적 문제해결을 성공적으로 해낼 수 있는 인재양성에 더욱 정진할 때이다.

본 연구는 이러한 시점에서 computational thinking(이하 CT)가 바로 [정보] 교과의 S/W 교육으로 연결되는 것이 자연스러운 지, 수학교육에서 CT의 사용이나 개발 가능성은 없는가에 관한 문제의식에서 출발하였으며, CT의 정의와 그 구성 요소와 수학교육에서의 CT의 의미인지, 또 수학교과에서 CT교육의 가능성 여부와 이를 위한 선행 조건을 탐색할 목적으로 수행되었다.

II 장에서는 컴퓨터공학, 학교교육, 수학 · 과학 교과에서의 논의되는 CT의 성격과 요소를 관련 연구를 통해 살펴보고, III 장에서 CT, 컴퓨팅(컴퓨터활용), 수학교육 사이의 관계를 논의하고 세 영역이 어떤 점에서 차이가 나는 지에 주목하며 요약하였으며, IV 장은 선행연구를 바탕으로 학교수학에 CT 교육을 포함시키는 방식과 이를 위한 조건 등을 제시하였다.

II. 계산적 사고(CT)와 수학 교육

본 장에서는 computational thinking(계산적 사고)의 정의와 그 구성 요소를 살펴보고자 한다.

1. 계산적 사고(CT)[2)]

Papert(1980)는 'computational thinking'을 그의 저서 *Mindstorms*에서 한 차례 컴퓨터의 역량과 관련시켜 사용(p. 182)한 바 있다. 그는 컴퓨터의 발달과 보급이 전통적인 학교 교육에 위협이 될 수 있으며 전통적 교실과 수업에 대안이 교육적으로 강력한 컴퓨터 환경 구축을 통해 제시될 수 있다고 하였다. 그는 당시 컴퓨터의 성능이 너무 원시적이어서 재미있고 흥미진진한 활동이 필요로 하는 역량을 갖추고 있지 못하여 'computational thinking'과 일상생활에 통합 방식이 충분히 개발되지 않았다고 언급하였다(p. 182). Papert의'computational thinking'은 컴퓨터 환경에서의 효율적인 사고를 의미하는 듯하다.

'Computational thinking(계산적 사고, CT)'이 최근 학계의 주목을 받게 된 것은 Wing(2006)에 의해서이다. Wing은 미국수학회 Communication에 실린 기고문에서 계산과 컴퓨터의 확산으로 인하여 CT가 컴퓨터 과학자 뿐 아니라 모든 사람이 갖추어야 할 기본 기능이 되었다며 아동이 기본적으로 갖추어야 능력으로 3R(읽기, 쓰기, 셈하기)에 CT를 추가해야 한다고 주장한다.

Wing의 CT는 '컴퓨터 과학의 기본 개념을 끌어들여 문제를 해결하고 체제를 설계하며, 인간의 행위를 이해하는 방식'(p. 33)이며, 인간이나 기계, 어느 것이든 수행주체와 관계없이 계산 과정의 역량과 한계를 기반으로 하는 사고[3)]이다. 그녀는 계산적 사고(CT)의 특징을 다음과 같이 몇 가지로 요약하였다(Wing, 2006).

• CT는 어려워 보이는 문제를 추상화나 분해를

2) 국내 문헌에서 computational thinking을 '계산적 사고' 또는'컴퓨팅 사고력'으로 번역하고 있는데 본 연구에서는 '계산적 사고'로 번역하였다. 그 이유는 컴퓨팅(computing)은 computer 사용[조작]의 의미가 강하며, 또 영어로 'computing thinking'이 아니라 'computational thinking'로 지칭된 용어를 '컴퓨팅(computing) 사고'로 번역하는 것이 적절하지 않다고 판단하였기 때문이다. (참고: 교육부 자료에는 컴퓨터 S/W 코딩 교육과 연계되어 '컴퓨팅 사고력'으로 지칭되고 있음.)

3) 인용문의 밑줄은 필자가 강조하기 위하여 친 것임.

사용하여 우리가 알고 있는 문제로 재공식화 하는 것이다.

- CT는 재귀적으로 사고하는 것이다.
- CT는 크고 복잡한 과제의 공략, 또는 크고 복잡한 시스템을 설계하는데 추상화와 분해를 사용한다.
- CT는 중복, 피해방지, 오류수정을 통해 최악의 시나리오를 방지, 보호, 회복과 관련하여 사고하는 것이다.
- CT는 답을 찾는데 발견적 추론을 사용한다. 이는 불확실성 안에서 계획하고 학습하고 일정을 짜는 것이다. 시간과 공간, 과정 역량과 저장 공간 사이에서 적정점을 찾는 것이다.

Wing은 CT를 프로그래밍이 아닌 개념화(Conceptualizing, not programming), 진부한 기술이 아닌 본질적인 기술(Fundamental, not rote skill), 컴퓨터가 아닌 사람의 사고방식(The ways that humans, not computers, think), 수학적 사고와 공학적 사고의 보완과 결합(Complements and combines mathematical and engineering thinking)이며, 가공물이 아닌 아이디어(Ideas, not artifacts)이며, 모든 사람에게 어디서나(For everyone, everywhere) 필요한 사고라고 하였다(Wing, 2006, pp. 34-35).

즉, Wing에게 'computational'은 사람이 컴퓨터를 활용하는 방식을 따른다는 의미이며, 컴퓨터를 사용한다는 의미는 아닌 것이다.

2. 계산적 사고(CT) 교육

교육적 맥락에서 CT에 대한 정의는 매우 다양하다. 영국의 공영방송 BBC(2017)는 KS(Key Stage)3의 '컴퓨터 교과'에서 CT를 소개하면서, 문제해결을 위해 컴퓨터를 사용하기 전에 문제 자체와 해결방식을 이해해야 하는데 이 과제에 도움이 되는 기법이 CT라고 하였다. 그리고 CT의 핵심초석으로 분해, 패턴인식, 추상화, 알고리즘을 지목하고 있다.

- 분해-복잡한 문제에 체계를 보다 작게, 다루기 쉽게 분해하기
- 패턴인식-문제 사이에, 문제 안에서 유사성 찾기
- 추상화-부적절한 부분은 무시하고 중요한 정보에 초점두기
- 알고리즘- 문제의 단계별 해, 또는 문제해결에 적용할 규칙을 개발하기

학교 교육에서 CT의 성격을 규명하고 CT 교육과 관련하여 여러 연구가 수행되었다(Barr, Harrison, & Conery, 2011; Grover & Pea, 2013; Barr & Stephenson, 2011; Yadav, Hong, & Stephenson, C, 2016, etc.). Barr, Harrison, & Conery(2011)는 미국과학재단(NSF) 지원으로 수행한 연구에서 계산적 사고(CT) 개념의 성격을 광범위하게 논의하였으며, 그 결과 교육자들에게 CT를 문제해결 과정으로 규정하고 CT의 조작적 정의를 다음과 같이 제시하였다. 이를 요약하면 다음과 같다.

- CT는 문제해결과정으로 다음을 포함한다. ① 풀이를 위해 컴퓨터나 다른 도구를 사용할 수 있는 방식으로 문제를 공식화하기, ②자료를 논리적으로 조직하고 분석하기, ③모델과 시뮬레이션 등 추상화를 통해 자료를 표현하기, ④산술적 사고(일련의 순서 단계)를 통해 풀이를 자동화하기, ⑤가장 효율적이고 단계와 자원의 효과적인 조합의 달성을 목적으로 가능한 답을 규명, 분석, 해석하기, ⑥이 문제해결 과정을 광범위한 문제에 일반화, 전이하기.

또한 CT 향상에 필요한 성향과 태도를 다음과 같이 제시하였다.

- CT를 지원하고 향상시키는 여러 성향과 태도는 다음을 포함한다. ①복잡성을 다루는데 자신감, ②어려운 문제를 다루는데 인내심, ③애매성을 용인하기, ④열린 문제를 다루는 능력, ⑤공동의 목적이나 해법을 위해 다른 사람들과 소통하고 협력하기.

Grover & Pea(2013)는 프로그래밍을 “CT 관련 인지적 과제를 지원하는 중요한 도구일 뿐 아니라 계산능력의 증거”(p. 40)라며 CT 교육에서 프로그래밍의 역할을 중시하였다. 이들은 계산적 사고(CT)가 패턴을 정의하고 특수 사례에서 일반화하는 ‘추상화’가 중심이라는 점에서 다른 사고유형과 구별된다고 하였다. Grover & Pea가 교육과정의 기초로 제시한 CT의 구성 요소는 다음과 같다.

- 추상화와 패턴일반화
- 정보의 조직적 처리
- 기호체계와 표현
- 통제 흐름의 알고리즘개념
- 구조적인 문제분해
- 무한반복, 재귀적, 평행적 사고
- 조건 논리
- 효율성과 수행제한
- 디버깅과 체계적 오류 추적

K-12 수준의 학교교육에서 CT는 대체로 독자적 교육과정으로 운영된다(Gadanidis, 2017). 그러나 CT가 일반적 문제해결 뿐 아니라 전통적인 교과 영역에서의 문제해결에 적용될 수 있으며, 학교교육에서 CT 개념은 교과와 통합적으로 접근되어야 한다는 주장이 점차 제기되고 있다(Qualls & Sherrell, 2010; Barr & Stephenson, 2011; Yadav, Hong, & Stephenson, 2016, Weintrop, Beheshti, Horn, Orton, Jona, Trouille, & Wilensky, 2016).

Yadav et al.(2016)은 ‘알고리즘, 추상화, 자동화’를 CT의 핵심요소로 보았으며, 이는 모든 학생에게 필요한 사고라고 하였다. 그런데 컴퓨터 교과에서 프로그래밍 교육으로는 모든 학생에게 CT 아이디어를 가르칠 수 없고, 또 초·중등의 전통적 교과에서도 CT가 구현될 수 있으며 그 영향이 긍정적이라며 다른 교과 영역을 통한 CT 교육을 주장하였다. 한 예로 초등학교 6학년 학생을 대상으로 Scratch를 사용한 수학 수업에서 긍정적인 효과가 보고된 사례(Calao, Moreno-Le´on, Correa, & Robles. 2015)를 예시하였다. Brennan & Resnick(2012)은 CT를 세 차원-계산적 개념(concepts) 차원, 실천(practices) 차원, 관점(perspectives) 차원-에서 구분하였다(Lye, & Koh, 2014).

Barr & Stephenson(2011)은 CT를 “컴퓨터로 시행이 가능한 방식으로 문제를 해결하는 접근”(p.51)으로 규정하고, 학생들은“추상화, 재귀, 무한반복 등 개념”을 사용하여 자료를 처리하고 분석하며 실제적 가상 인공물을 만들 수 있다고 하였다. 또 CT는 자동화와 전이가 가능하며 모든 교과에 적용될 수 있다고 하였다. 이들은 자료수집 및 분석, 자료표현, 추상화, 알고리즘과 절차, (도구사용을 통한) 자동화, 변형(풀기), 시뮬레이션을 핵심적 CT 개념과 능력으로 제시하였으며, 이들 핵심 개념과 능력별로 K-12 수준의 컴퓨터, 수학, 과학, 사회, 언어 등 교과 수업에서 구현사례를 표로 예시하였다(p. 51).

<표 II-1>은 교육적 맥락에서 CT를 다룬 연구 또는 문서에 나타난 대상 학년 또는 교과, 그리고 CT 구성요소를 표로 요약한 것이다. <표 II-1>에 의하면, CT는 컴퓨터(또는 정보) 교과를 통한 교

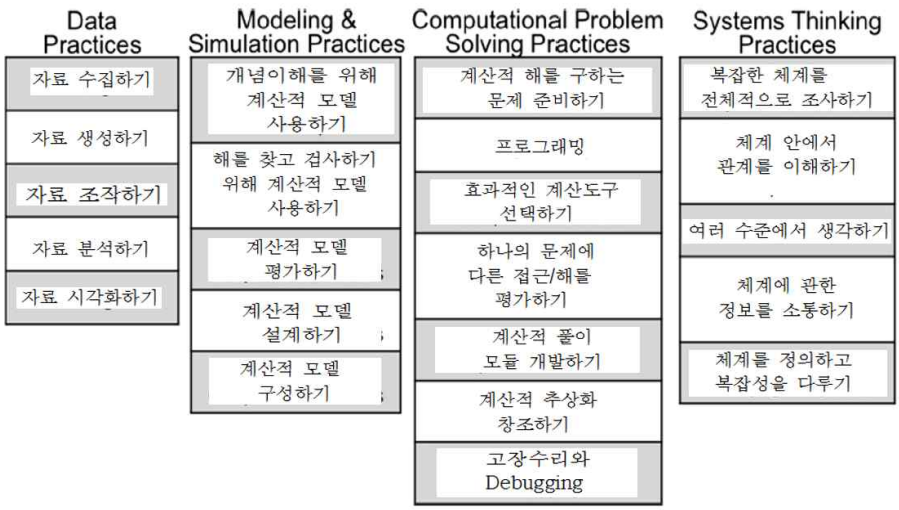
<표 II-1> CT 교육관련 문헌에 나타난 논의 수준과 CT의 구성요소

출처	Barr, Harrison, & Conery (2011)[4]	Grover & Pea (2013)	BBC (2017)	Angeli, et al. (2016)	Yadav et al. (2016)	2015 교육과정
논의 수준	K-12	K-12	컴퓨터 교과	K-6	K-12	정보 교과
구성 요소	• 문제 공식화 • 자료 조직·분석 • 추상화→자료 표현 • 풀이 자동화 • 최적해 찾기 • 일반화 및 전이	• 추상화와 패턴일반화 • 조직적 정보 처리 • 기호체계와 표현 • 알고리즘 • 문제 분해 • 무한반복,재귀,평행 적사고 • 조건 논리 • 효율성과 수행제한 • 디버깅 & 오류 추적	• 분해 • 패턴인식 • 추상화 • 알고리즘	• 추상화 • 일반화 • 분해 • 알고리즘 • 디버깅	• 알고리즘 • 추상화 • 자동화	• 추상화 • 자동화(프로그래밍) • 창의·융합

육으로 다루어지기도 하지만 K-12 전 학년에서 다른 교과를 통한 교육으로 CT교육의 적용 범위가 확장되어 논의되고 있음을 알 수 있다.

교육적 맥락에서 언급되는 CT는 분해를 통한 문제의 공식화 등 '문제해결에 도움이 되는 접근 방식', 또는 '컴퓨터로 시행이 가능한 방식으로 문제를 해결하는 접근'으로 요약할 수 있으며, CT의 요소로 무한반복, 재귀, 디버깅 같이 컴퓨터에서 중요하게 사용되는 개념이 포함되며, 공통적으로 거론되는 주요 CT 요소는 추상화, (프로그램을 통한) 자동화, 알고리즘, 최적화이다.

CT가 수학적 또는 문제해결과 어떻게 관련이 되는 지, 어느 정도의 프로그래밍이 필요한지, CT가 항상 컴퓨터를 필요로 하는 지에 관해서는 지속적으로 논의할 필요가 있다. 그러나 모든 교과에서 CT 교육이 가능하다는 연구 결과에도 불



[그림 II-1] 수학·과학 교과에서 실천 차원의 CT 분류
(Weintrop et al., 2016, p. 135, 그림2)

4) 이 연구에서 CT의 정의적 요소를 구별하여 제시하였으나 이 표에는 생략하였음.

구하고, 수학교과 맥락에서 이를 적용하기 어려운 이유는 CT 개념의 정의가 명확하지 않고 포괄적으로 논의되기 때문이다(Weintrop, et al., 2016).

Weintrop et al.(2016)은 중등학교 수학교육에서 계산적 사고의 정의와 성격 규명을 STEM (Science, Technology, Engineering, & mathematics)[5] 교육 또는 과학 교과와 연계하여 논의하였다[6]. Weintrop et al.(2016)은 문헌을 통해 CT의 핵심 기능과 실천 조사, 고등학교 수학·과학교과에서 사용하는 CT 도입을 위한 활동 분석 및 범주화, 교사들 대상으로 예비 검증, STEM 전문가 인터뷰 등의 절차를 거쳐, 수학 및 과학 수업에 CT 적용을 위한 실천 유목을 정의하였다.

Weintrop et al.(2016)은 실천 차원에서 도출한 네 개의 주요 범주(자료, 모델링과 시뮬레이션, 문제해결, 체계사고)와 범주별로 5~7개의 실천(practice) 목록이 포함된 수학·과학에서의 CT 분류를 개발하였다. [그림 II-1]은 이 분류를 도해한 그림으로, 수학과 과학의 맥락에서 CT의 성격을 보다 분명하고 구체적으로 규명하고 있으며 각 범주별 실천 목록은'실행에 옮길 수 있는, 수업에 바로 사용할 수 있는 CT'의 정의(p. 128)로 볼 수 있어 수학교육의 맥락에서 CT 활동 개발의 준거가 될 수 있을 것이다.

Weintrop et al.(2016) 연구에서 CT의 네 범주에 포함된 실천은 다음과 같다.

- 자료 실천
  자료의 수집, 분석, 조작, 시각화와 관련한 실천
- 모델링과 시뮬레이션 실천
  계산적 모델 창조, 그 강점과 제한점 평가, 학습의 도구로서 모델 활용과 관련한 실천
- 계산적 문제해결 실천
  모든 수준에서의 사고, 시스템 내에서 관계의 이해, 전체로서 시스템에 관한 추론과 관련한 실천
- 시스템 사고 실천
  복잡한 체계를 전체로 보기, 체계 내애서 관계의 이해, 여러 수준에서 사고, 체계에 관한 정보 소통, 체계의 정의와 복합성 다루기 등의 실천

이 프로젝트는 모든 고등학생들이 컴퓨터가 만연한 미래를 살아 갈 모든 학생들에게 고등학교 수학과 과학에서 계산적 사고의 향상을 목표로 하여 STEM 분야에서 교육과정과 평가를 구조화하기 위한 CT 분류를 개발하고, 수업과 활동 자료, 평가자료, CT 지도를 위한 전문성개발 프로그램을 지원하는 홈페이지를 운영하고 있다 (http://ct-stem.northwestern.edu/).

이 홈페이지에는 고등학교 수학 과학 수업에서 CT 실천을 다루는 수업 계획 24개를 제시하고 있는데 그 중 5개가 수학 수업과 관련된 것으로 그 수업 제목은 다음과 같다.

<표 II-2> CT를 다루는 수학 관련 수업 목록

수업제목	과목
실제 방사능 자료 사용 수학적 모델링	화학&수학
Lattice 나라: 다각형 탐구	수학
Lattice 나라: lattice 삼각형 탐구실천	수학
전기차에서 회생제동의 물리학	수학&물리
미국 인구조사의 주택자료 Sine 함수로 모델링하기	수학

(출처: http://ct-stem.northwestern.edu/)

5) 우리나라에서는 과학-테크놀로지-공학-수학을 뜻하는 STEM에 Art를 추가하여 융합교육을 STEAM으로 약칭한다.

6) Wing(2008)은 수학은 논리적 구조나 수학적 대상의 모델 능력에서 계산적 사고와 연결된다고 하였다.

이 프로젝트에서 고등학교 교사 대상 전문성 개발 프로그램의 목표가 학생들에게 다음을 할 수 있게 하는 것이라 제시하였다.

- 기존의 CT 도구로 STEM 학습 향상
- (컴퓨터 사용 또는 사용하지 않는) 계산적 문제 분석 안내
- 실세계 자료와 문제를 분석하도록 알고리즘과 모델의 개발 기원
- CT-STEM 경력이나 연구에 참여 기회
- STEM에서 CT 이해 발전

Weintrop et al.(2016)에서 고등학교 수학교과에서의 CT 적용은 STEM 교육의 맥락에서 수학교과 내용의 융합을 의미하며, 여기에서 CT는 문제해결을 위한 컴퓨터 활용([그림 II-1] 개념이해, 풀이와 검사에 컴퓨터 사용하기)도 포함한다.

4. 국내의 CT 교육 현황

최근 국내에서 CT 관련 연구는 주로 컴퓨터 전공 교수들에 의해 수행되었다(전영국, 2015; 김석전, 전용주, & 김태영, 2015; 권정인, & 김재현, 2016; 최숙영, 2011). 권정인 & 김재현(2016, p.48)은 CT를 문제 해결에의 단순한 컴퓨팅의 활용이 아닌 창의·융합적 차원에서 하는 사고라 서술하였으나 내용은 컴퓨터 활용과 직접적으로 관련된다.

이영준 외(2014)는 초·중등학교에서 CT 도입을 위한 기초 연구[7]에서, 계산적 사고력을 "문제해결에 컴퓨터나 다른 도구를 사용할 수 있도록 문제를 구성하기, …(중략)…, 이러한 문제해결과정을 다양한 문제들로 일반화하고 전환하기를 포함"(이영준 외, 2014, p.33)하는 것이라 기술하였다. 즉 계산적 사고를 '컴퓨팅 시스템의 역량을 활용하여 해결하고자 하는 문제를 효과적이고 효율적으로 해결할 수 있는 절차적 사고 능력'(p. 40)이며 '컴퓨터나 다른 도구 사용'에 필요한 능력으로 보았다.

창의적 문제해결력을 갖춘 미래인재양성에 대한 국가적 관심은 소프트웨어 운영지침(교육부, 2015c)을 발표하는 등 코딩교육의 강화로 나타났으며, 2018년부터 초·중등학교에서 S/W 교육의 의무화하도록 되어 있다(미래창조과학부·교육부, 2015).

> 2020년까지 초·중등생에 대한 SW교육을 통해 창의적 아이디어를 SW로 구현할 수 있는 문제해결력을 갖춘 '미래형 창의인재'를 양성하고, … (중략)… SW교육은 창의적 아이디어를 SW로 구현하는 사고력 교육이며… (중략)… 문제해결력, 컴퓨팅사고력 개발과 학생이 체험할 수 있도록 하는 교과서를 개발하여 보급하며, … (미래창조과학부·교육부, 보도자료, 2015)

2015 교육과정(교육부, 2015b)은 '컴퓨팅 사고력'을 [정보] 교과에서 추구할 역량으로 명시하였으며, 그 성격을 다음과 같이 기술하고 있다.

> '컴퓨팅 사고력'은 <u>컴퓨터과학의 기본 개념과 원리 및 컴퓨팅 시스템을 활용하여 실생활과 다양한 학문 분야의 문제를 이해하고 창의적으로 해법을 구현하여 적용할 수 있는 능력</u>을 말한다[8]. '컴퓨팅 사고력'은 '추상화(abstraction) 능력'과 프로그래밍으로 대표되는 '자동화(automation) 능력', '창의·융합 능력'을 포함한다. 추상화는 문제의 복잡성을 제거하기 위해 사용하는 기법으로 핵심 요소 추출, 문제 분해, 모델링, 분류, 일반화 등의 방법으로 이루어진다. 추상화 과정을 통해 도출된 문제 해결 모델은 프로그래밍을 통해 자동화된다.(p.96)

---

7) 이 보고서 제목에는 computational thinking(CT)로 썼으나 본문에서는 이를 컴퓨팅사고력으로 번역하였음.
8) 밑줄은 본 연구자의 것임.

위 진술의 밑줄 부분에 따르면, CT는 컴퓨터의 개념, 원리, 프로그래밍을 통한 다양한 분야의 문제해결 능력을 의미한다.

우리나라에서는 CT가 컴퓨팅 사고로 번역되면서, CT가 컴퓨터 교과 관련 사고일 뿐 수학 교과와의 관련성이나 문제해결로서의 CT 자체는 그리 주목을 받고 있지 못하다.

이것은 CT를 '수행 주체와 관계없이' 계산 과정의 역량과 한계를 기반으로 하는 사고라는 Wing의 설명과는 배치되는 입장이며, 최근 전통적 교과에서의 문제해결을 통해 CT 교육이 가능하다며 CT를 K-12 모든 학년 수준에서 다른 교과를 통해 구현하려는 국외 동향과는 간극이 있어 보인다.

III. CT, 컴퓨팅, 수학교육,

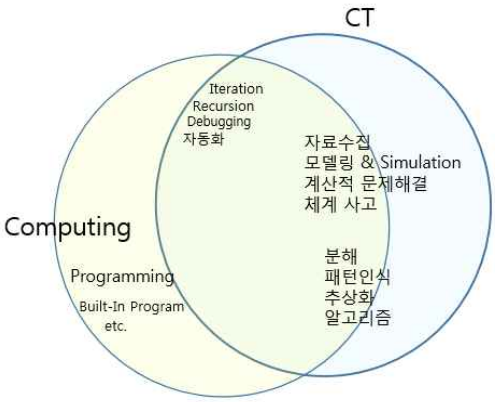
1. CT, 컴퓨팅, 수학교육

본 연구에서는 CT를 분해를 통한 문제의 공식화 등 '문제해결에 도움이 되는 접근 방식', 또는 '컴퓨터로 시행이 가능한 방식으로 문제를 해결하는 접근'으로 보기로 한다

선행연구를 바탕으로 CT, 컴퓨팅(computing), 수학교육 사이의 관계를 살펴보기로 한다. CT에 대한 선행연구의 정의가 다양한 차원에서 기술되고 있으며, 수학 교육 영역 역시 학년수준과 교육과정에 따라 다루는 부분에 차이가 있어 일반화가 어렵다. 그러나 요소 사이의 관계 기술에 대략적이라도 도식화가 유용하다고 판단하여 세 요소 사이에 도식화를 시도하였다.

가. CT와 컴퓨팅

CT는 컴퓨팅 환경을 기반으로 하는 사고이므로, 컴퓨터의 역량과 관련이 있다. CT는 컴퓨팅 도구의 명령세트, 자원의 제약, 운용환경의 영향을 받는다. 그러므로 CT를 ICT(Information and Communication Technology) 기기나 컴퓨팅과 전혀 별개로 생각하기는 어렵다. LOGO의 초기 개발자인 Pappert(1980)가 CT를 일상생활에 통합시킬 수 없는 이유를 당시 컴퓨터의 '원시적인'성능으로 인한 역량 부족으로 재미있는 활동을 할 수 없었다고 한 것도 같은 맥락이다.



[그림 III-1] 컴퓨팅과 CT

[그림 III-1]은 컴퓨터 활용이 곧 CT의 사용과 동일한 것이 아니라는 것을 나타낸다. CT는 문제해결과 지식표현을 위한 계산도구를 사용하는데 필요한 열쇠(Yadav et al., 2016)이며, 단순한 컴퓨터 소양과는 다르다. 즉, CT는 컴퓨팅이 문제해결과 표현도구가 될 때 필요한 사고이다. 컴퓨팅이 항상 CT를 사용하는 것은 아니다. 기계적인 프로그래밍이나 프로그램에 내장된 기능을 기계적으로 사용하여 문제를 해결하는 경우 CT를 사용하는 것과 차이가 있다.

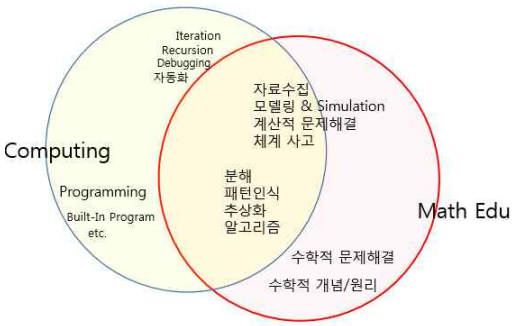
또한 CT의 개발과 활용은 실제로 컴퓨팅 도구 사용에 이르지 않는 환경에서도 가능하다. <표 II-1>에 포함된 CT의 일부 요소, 예를 들어 문제를 분해하고 풀이에 적합하도록 공식화하는 능

력은 컴퓨팅 도구 활용 이전에 필요한 사고이며 일반적인 문제해결에도 유용하고 적용이 가능한 사고이다.

나. 컴퓨팅과 수학교육

[그림 III-2]는 컴퓨팅과 수학교육 사이의 사이를 나타낸 것이다. 우리나라 수학 교육에 컴퓨터 도입이 최초로 언급된 것은 제 6차 수학과 교육과정으로, '복잡한 계산이나 문제 해결력 향상 등을 위하여 계산기나 컴퓨터를 활용할 수도 있다'(교육부, 1992, p. 42)는 기술로 컴퓨팅 도구 사용을 허용하였다. 이후 수학과 교육과정에서도 컴퓨팅 도구는 교수·학습과정에서 전통적 '수학교육'의 보조 수단으로 사용이 허용되고 있다.

> 교수·학습 과정에서 "계산 능력 배양을 목표로 하지 않는 경우의 복잡한 계산 수행, 수학의 개념, 원리, 법칙의 이해, 문제 해결력 향상 등을 위하여" 계산기, 컴퓨터, 교육용 소프트웨어 등의 공학적 도구와 다양한 교구를 활용한다." (교육부, 2011, p.55 ; 2015a, p.39)



[그림 III-2] 컴퓨팅과 수학교육

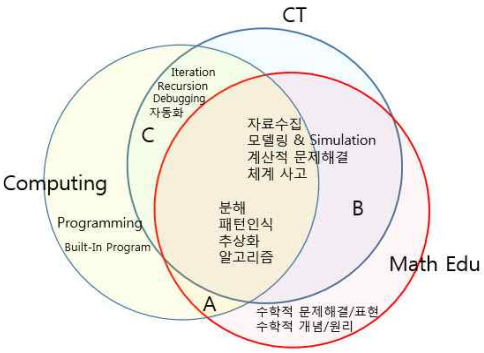
또한 평가에서는 "평가 내용과 방법에 따라 계산기, 컴퓨터, 교육용 소프트웨어 등의 공학적 도구와 다양한 교구를 이용" 할 수 있는 기회를 '제공'(교육부 2011, p. 56), 또는 '제공할 수 있게'(교육부 2015a, p. 41) 하였다. 그러나 대부분의 수학 수업은 컴퓨터 사용과 무관하게 진행되고 있고 컴퓨팅의 핵심요소인 무한반복이나 재귀(recursion)도 수학교육에서 다루기는 하나 비중있게 다루어지지 않고 있어 [그림 III-2]에서 수학교육에 인접한 컴퓨팅 영역에 표시하였다.

물론 계산이나 시각화 등으로 컴퓨팅 도구가 전통적인 수학교육을 돕는 보조 수단으로 활용될 수 있다. 그러나 컴퓨팅 도구의 역량은 단순한 보조수단을 넘어 수학 교육을 더 깊이 있고 풍부하게 할 수 있다.

Gadanidis(2017, P. 133)에 따르면, "테크놀로지는 단지 인간의 의도를 위한 도구가 아니다". 테크놀로지는 인간의 사고를 지원하기도 하지만 때로는 방해하고 재조직하기도 한다. 이러한 입장이라면 수학교육에서의 컴퓨터 활용은 교육내용의 변화를 가져올 수 있다. 그리고 컴퓨터의 활용 방식을 따르는 사고가 CT라고 한다면 (Wing, 2006), 활용하는 컴퓨팅 기기의 명령세트나 자원의 제약, 활용 환경에 따라 수학교육의 내용과 교실에서의 컴퓨터의 활용 방식이 달라질 수 있다. 또한 학교 수학에서의 컴퓨팅 도구 사용은 환경 구비를 위한 재정 부담, 도구 활용을 위한 충분한 수업시수 확보, 수업에서 예상치 못했던 노이지(noisy)에 대한 대응을 위한 교사 역량 구비 등 쉽지 않은 여러 과제를 안겨준다.

다. CT, 컴퓨팅, 수학교육

CT, 컴퓨팅, 수학교육은 서로 연관되며 이들 사이에 공통 영역과 각기 고유한 사고 영역이 있다. [그림 III-3]은 이를 나타낸 것으로 (타 교과 관련 부분은 여기에서 논외로 하고) 수학교육 영역에서 CT와 관련하여 시사점을 얻을 수 있는 A, B, C 세 부분에 주목하고자 한다.



[그림 III-3] CT, 컴퓨팅, 수학교육

- A: 컴퓨팅 환경에서 이루어지는 수학교육이나 CT가 아닌 것을 다룸.
- B: 컴퓨팅환경이 아닌 수학교육에서 CT가 사용되거나 개발됨.
- C: 컴퓨팅환경에서의 CT이나 수학교육 영역에 포함되지 않음.

[그림 III-3]에서 A는 수학교육에 컴퓨팅 도구를 사용하지만 CT가 관련된다고 보기 어려운 경우로, 예를 들면 복잡한 계산이나 대수조작으로 답을 구하는 수단으로 간단히 컴퓨팅 도구를 사용하는 것이 이에 속한다.

[그림 III-3]의 B는 수학교육에서 CT가 사용 또는 개발되지만 컴퓨팅 도구를 사용하지 않는 경우가 이에 해당한다. 복잡한 문제를 분해하여 수학적으로 해결하는 수행이 이에 속한다. 지필환경에서 복잡한 문제해결을 위한 분해, 모델링, 알고리즘의 적용하는 것이 속한다.

[그림 III-3]의 C는 컴퓨팅 환경에서 구현이 용이하고 CT를 요구하는 내용이고 수학적으로도 의미가 있으나 현재 우리나라 수학교육의 범위를 벗어난 경우가 C에 속한다. 예를 들면, 수학적 맥락에서 문제해결을 위하여 무한반복, 재귀를 활용한 프로그래밍, 디버깅 상황, CBR 등으로 자료를 수집과 분석하여 해를 도출하는 활동 등은 수학적 활동이기도 하지만 현재 수학 교육과정의 범위를 벗어난 것들이다.

수학교육에서 CT를 고려한다고 할 때, 주목할 만한 영역이 [그림 III-3]의 C 영역이다. C 영역은 현재 수학교육에 포함되어 있지는 않으나 컴퓨팅 환경에서 가능한 CT이며, 컴퓨팅 환경이 아니면 다루기 어려워 수학교육과정에서 배제된 것일 수 있다.

재귀나 무한반복은 지필환경에서 다루기 어려운 CT 개념으로 컴퓨터 프로그래밍이나 엑셀환경에서 매우 유용하게 사용된다. 재귀(recursion)는 식이나 함수를 바로 그 자신을 사용하여 표현하는 방식으로 고등학교 수학에서 '점화식' 또는 '귀납적 정의'로 불리며 다루어져 왔다. 그러나 2011 교육과정부터 점화식' 용어가 삭제되고 '귀납적 정의' 또는 귀납적 표현으로 점화식 수열을 다루고 있으며, 몇 개의 특정 항은 구하되 그 이전까지 다루던 일반항(a_n)을 다루지는 않도록 교육과정에 약화시켜 다루고 있다[9]. '알고리즘과 순서도'도 2011 교육과정에서 삭제되었는데[10], 전체 학습량 감축이 반영된 결과라 할 수 있겠으나 두 개념 모두 컴퓨팅 환경에서는 자연스럽게 학습이 가능하지만 지필환경에서 이들의 의미와 효용성의 체감 역시 쉽지 않은 CT 개념이다. 이 경우에 컴퓨팅 환경 유무가 수학교육과정의 경계를 설정하는 한 요인이 되는 셈이다.

9) 수열 단원의 약화와 학습량 감소의 개정 방향이 반영된 결과로 판단됨.

10) 순서도는 문제해결을 강조와 함께 제 4차 수학교육과정에, 알고리즘은 5차 수학교육과정에 처음으로 도입되었으며, 모두 2011 수학교육과정에서 삭제되어 현재 교육과정에서 다루지 않는다. '알고리즘과 순서도'가 수학적 문제해결 대한 일반적 접근 방식이긴 하나 프로그래밍을 배제한 상황에서 지도의 의의와 유용성을 찾기 어려웠던 것으로 판단된다.

IV. CT와 수학교육의 통합

1. 컴퓨팅 환경에서의 수학교육

수학교육에 CT를 통합하여 다루는 것은 문제해결과 지식의 표현에 컴퓨팅 환경을 이용할 수 있어야 한다.

수학적 문제해결을 위해 복잡한 요소를 분해하고 공식화 하는데 필요한 사고도 CT로, CT의 개발과 활용은 실제로 컴퓨팅 환경이 아니라도 가능하다. 그러나 문제해결을 지필환경으로 한정하는 경우, 계산 역량의 한계 때문에 실질적인 자료를 다루지 못하며 또 적절한 수학적 모델을 찾았다고 해도 답을 구하지 못할 가능성이 크다. 그러므로 학생들로 하여금 CT를 충분히 적용하여 실질적인 문제해결에 이르게 하려면 컴퓨팅 도구 사용이 가능한 수학 교실 환경 조성이 필요하다[11]. 수학교육에서 CT에 주목하여 문제해결과 사고 교육을 컴퓨팅 환경에서 시행할 필요가 있으며 수학교육에서 CT의 통합은 문제해결과 지식의 표현을 위해 컴퓨팅 도구를 사용(Yadav et al., 2016)하게 하는 것에서 출발한다.

가. 수학교육에서 코딩

컴퓨터 교과에서 프로그래밍(코딩) 교육으로 CT의 일반 요소를 교육할 수 있을 것이다. 그러나 대부분의 문제 상황은 맥락과 밀접히 연관되어 있기 때문에 수학적 문제해결이나 탐구와 관련한 프로그래밍은 수학 교과에서 다루는 것이 효과적일 수 있다. 컴퓨팅 환경은 피드백을 통한 즉각적인 사고의 평가, 분석적 사고를 통한 디버깅(오류수정), 실제적인 문제해결 결과, 지필환경에서 다루기 힘든 재귀(Recursion), 반복(Iteration) 등 개념을 이해하고 활용하는 기회를 제공한다. $n!$의 재귀적 표현 $n\times(n-1)!$이 $n\times(n-1)\times...\times1$을 의미하며 프로그래밍 환경에서 전자의 유용성과 역량을 인식하고 수학적 문제해결에 이를 활용하는 경험은 수학교육과 CT의 통합으로 가능한 결과이다.

알고리즘은 컴퓨터가 자료를 처리하는 방식으로 지필 계산과 차이가 있다. 프로그래밍은 컴퓨터가 이해할 수 있도록 번역하는 것이며 그 과정은 알고리즘을 이해하고 적용하는 것이다. 예를 들어, 지필환경에서는 '$17\div5=3\cdots2$'를 직관적으로 17에서 단번에 $5\times3=15$를 빼면 2가 남으므로 '몫이 3, 나머지가 2'인 것으로 계산하지만, 이를 몫과 나머지를 구하는 프로그래밍을 작성하려면, 17에서 나머지가 5보다 작게 나올 때 까지 5를 계속 빼면서 뺀 횟수와 그때 나머지를 기록하는 과정으로 알고리즘을 분해하고 이해하는 것이 필요하다. 알고리즘은 문제해결 방식을 단계적으로 분석에서 나온다.

수학교과와 CT를 통합할 때 사용하는 프로그래밍 언어는 가능한 대로 코딩 방식이 간편해야 하며 사용 방법을 익히는 것이 부담스럽지 않아야 한다. 그런 의미에서 LOGO나 scratch는 프로그래밍 방법이 복잡하지 않고 다양한 수준에서 사용이 가능하여 수학교육에 적합한 언어로서 CT 교육 연구(Calao, et al., 2015; Gadanidis, 2017)에 사용되기도 하였다. LOGO는 교육용 프로그래밍 언어로 개발된 절차적 언어로 일반적인 컴퓨터 언어와 달리 작은 단위의 모듈(procedure)로 나뉘어 작동하며, 오류에 대해 즉각적인 피드백이 일상적인 언어로 주어지는 등 여러 수준에서 수학교육을 위한 언어로 활용될 수 있다(Abelson, 1981; Cuoco, 1990, Hoyles, & Noss, 1992; 김화경, 2015). 스프레드시트(Excel)는 본격적인 프로그래

11) 수학교육에서 CT를 컴퓨터 사용과 관련이 없이 문제해결과 시뮬레이션을 개발 또는 사용할 수 있으나 여기서는 컴퓨팅 환경에서의 통합에 주목하기로 한다.

밍 언어는 아니나 접근이 용이하며 셀을 채우는 데 수식 관련하여 간단한 프로그래밍 기법이 사용되어 수학적 개념 학습이나 문제해결에 유용한 CT를 개발하고 구현할 수 있는 S/W이다.

나. 수학교육에서 CT 통합

컴퓨팅 환경이 사고력 중심의 수학교육을 지원할 때 컴퓨터 활용이 긍정적으로 평가할 수 있으며(나귀수, 2010), 컴퓨팅 도구는 코딩 뿐 아니라 소프트웨어를 사용하여 패턴을 탐구하고 모델링의 결과를 찾아내는 활동을 통해 CT를 수학적 문제해결 또는 개념 및 원리 학습을 도울 수 있다(장경윤, 황우형, & 이중권, 2001; 이화영, & 장경윤, 2015). 컴퓨팅 환경에서 수학 수업을 할 때, 컴퓨터가 단순한 보조도구가 아니라 CT 교육의 도구가 되게 하려면 CT가 필요한 상황을 문제로 제공하는 것이 필요하다.

수학적 맥락에서의 문제해결을 컴퓨팅 환경이 지원되지 않기 때문에 내용을 축소시키거나 자료를 가공하는 일은 적절하지 않다. 수학 수업에서 지필계산이 가능하도록 실제 자료를 인위적으로 가공하여제공하는 일은 학생들의 풀이를 도울 수 있다. 그러나 이것 때문에 학생들은 수학문제의 답은 간략한 형태로 나온다는 잘못된 신념을 갖게 되어, 실질적인 문제에 수학을 적용해야 할 때 실패하거나 문제해결 자체를 포기할 수 있다.

컴퓨팅 환경에서 CT의 통합은 다양한 방식의 해법을 가능하게 하므로 반드시 개념을 먼저 다루는 수업 계열에 변화가 가능하다.

다. STEAM 맥락에서 수학교육과 CT의 통합

Weindrop et al.(2016)은 CT를 수학·과학 수업에서 다루는 것이 CT와 두 교과 사이에 상호관계를 설정하고, 모든 학생들의 실질적인 관심사를 다룰 수 있으며, 학교 수학·과학을 이 분야의 전문적 실천과 맥을 같이 하는 이점이 있다고 하였다. 그리고 과학 수학교과에서 CT를 통해 STEM 교육이 가능하고 미래 사회에서 이 분야에 실질적 견해를 가지고 준비시킬 수 있다고 하였다.

수학교과에서 CT를 융합교과(STEAM) 맥락 에서 다룰 수 있도록 학생들에게 실제적인 문제를 제시하고 학생들은 수학과 학문 영역의 경계에서 문제 상황을 조망하고 해결을 위한 계획을 세우고 적절한 도구를 선택하여 해법을 찾고 결과를 반성하게 함으로써 학생들은 실질적인 문제해결자로서 경험할 수 있게 될 것이다. 최근 고호경 등(2016)이 수학 중심의 STEAM 교육 자료를 일부 개발하여 발표한 바 있다. STEAM 맥락에서 수학과 CT의 통합을 위하여 학생 수준에 적절한 실질적인 활동 개발하는 것이 가장 큰 과제라 할 수 있을 것이다.

2. CT 통합을 위한 제 조건

가. CT 통합을 고려한 수학교육과정

컴퓨팅 환경에서 CT를 고려한 수학 활동은 자연스럽게 학생들에게 교육과정 범위를 넘나들게 한다. 그러므로 컴퓨팅 환경에서 수학적 문제해결로 CT를 활용 또는 개발하려고 한다면 수학교육과정에 대한 논의가 불가피하다. 컴퓨팅 환경에서 수학적 문제해결을 할 수 있는 과제의 제시, 적절한 컴퓨팅 기기 선택과 환경의 구축, 그리고 수학교육과정이 이를 지원할 수 있도록 세심한 준비와 논의가 필요하다.

수학교육의 내용과 답과 풀이 방식의 허용 범위와 사고의 확장이 필요하다. 학생들은 교육과정에 제시된 수학적 방법으로 해를 얻지만 컴퓨팅 환경은 같은 상황에 대하여 다양한 접근을 가능하게 한다. 컴퓨터가 기기 내부 알고리즘으로 단

번에 근사 해를 제공하기도 한다. 컴퓨팅 환경에서 학생들 나름대로 탐구를 통해 근사 해를 구할 수도 있으며, 문제의 분해, 추상화 모델링의 과정을 중학생이 미적분 없이 컴퓨팅 기기로 최적 해를 구한 경우, 이를 어느 정도로 허용할 것인가 하는 등 교육과정 과 관련하여 논의가 필요하다.

컴퓨팅 환경에서의 문제해결 접근방식이 지필 환경과 다를 수 있으며, 동일한 개념이지만 표현 방식이 수학적 표현과 다른 경우도 있다. 그런 경우 수학 교과의 구성 때문에 관련 방식이나 표현을 체계적으로 다루지 않더라도 그 표현이 의미하는 바를 학생들의 이해를 돕는 수준에서 간략히 소개하는 것으로 문제해결에 학생들을 참여시킬 수 있다.

한 예로, 계산도구 활용 결과 2E-5를 답으로 얻었다면 학생들은 이를 0.00002으로 해석할 수 있어야 한다. 교육과정에서 2×10^{-5}라는 수의 과학적 표현을 다루지 않는 상황에서 이는 간단치 않은 문제이나 그저 무시하고 넘길 사안도 아니다. 휴대폰에 내장된 계산기에서도 흔히 발견되는 표현이기 때문이다. 학생들이 실제상황에 빈번히 나타나는 상황을 학교 수학에서는 다루지 않는 경우 수학 교과의 실용성을 의심할 수밖에 없게 한다.

컴퓨팅 환경은 다양한 접근을 가능하게 한다. 특정 방식을 지정하지 않았다면 다양한 접근 방식을 허용할 때, 학생들의 주도적이고 자발적 탐구가 활성화 될 수 있다. 학생들은 지필계산 참값을 구하는 것을 선호하지만 실제로 근삿값이 더 유용할 수 있으며, 지필계산으로 어려운 경우에는 CT를 활용한 풀이가 학생들이 접근 가능한 유일한 해일 수 있다.

나. 컴퓨팅 기기 관련 이해

자신이 다루는 컴퓨팅 도구에 대한 이해도 필수적이다. 도구를 잘 알지 못하면 수학적 사고와 문제해결이 원활하지 못하기 때문이다. 그리고 사용자가 주체가 되어 컴퓨팅 기기(ICT)를 도구로 사용하려면 도구로 발생되어야 한다(Trouche, 2003). 그 과정에서 교사는 도구의 가능성과 한계에도 주목해야 할 필요가 있다(Wing, 2006). 수업을 위한 프로그래밍을 위해 모든 기능을 한꺼번에 익히게 할 필요가 없으며 필수적인 최소 기능의 습득으로 CT나 수학적 사고에 집중할 수 있도록 적절한 도구의 개발과 선택이 필요하다.

다. CT 통합에 대한 수학교사의 이해

CT 능력을 수학 교과에서 다룰 것인가 하는 문제는 교육과정의 경계 설정과 관련된 문제로 합의가 필요한 부분이다. 그리고 CT 능력을 수학 교과에 포함시킬 때, 이를 수업에서 다루어야 하는 교사들의 이해와 인식이 필요하다. 컴퓨터 활용 없이 수학교과에서 CT에 주목한다고 하면 복잡한 문제해결이나 시뮬레이션이 강조되는 경우, CT의 핵심인 재귀는 컴퓨팅 도구 사용이 부가되는 경우, 그 중요성에 대한 교사들의 인식과 물리적인 시수 확보가 이루어져야 한다.

또 수학교육에서 컴퓨팅 도구의 사용은 수학교사의 역할을 확장시킨다. 교사주도로 간략히 수학의 핵심 개념과 원리를 가르치는 것 보다 학생 활동이 늘어나고 수업에 더 많은 조정과 더 많은 시간이 된다. 그러므로 CT를 수학교과에 통합하는 것에 대한 교사의 이해와 특별히 컴퓨팅 환경에서의 수업을 위한 교사의 준비가 필요하다.

Ⅴ. 요약 및 결론

미래는 정보통신 융합으로 이루어지는 제4차 산업혁명의 시대이며 컴퓨터나 인터넷 뿐 아니

라 인공지능, 로봇, 사물인터넷(IoT) 등 첨단의 기술공학은 이미 우리가 도처에서 가까이 만나는 환경요소가 되었다. ‘컴퓨터 과학의 기본 개념을 끌어들여 문제를 해결하고 체제를 설계하며, 인간의 행위를 이해하는 방식’으로 계산적 사고(CT)를 소개한 Wing(2006)은 이를 모든 사람이 갖추어야 하는 능력으로 3R에 추가해야 한다고 주장하였다.

본 연구에서는 computational thinking(CT)을 ‘계산적 사고’로 번역하였으며 본문에서는 CT로 약칭하였다. CT가 국내에서 컴퓨팅 사고력으로 번역되는 사례가 많은데, 컴퓨팅(computing)은 ‘컴퓨터를 사용하는’의 의미로 읽혀 ‘정보’ 교과의 전유물로 여겨지기 쉬우며, 실제로 컴퓨터를 사용하지 않아도 CT로 칭할 수 있기 때문에 본 연구에서는 계산적 사고로 번역하였다. 실제로 우리나라에서 CT에 대한 관심은 프로그래밍과 코딩(coding), <정보> 교과의 필수 교과 지정으로 나타났다.

21세기 미래인재가 갖추어야 할 필수 능력으로 간주되는 CT에 대한 정의는 <표 II-1>에 요약한 바와 같이 매우 다양하며 포괄적이다. CT의 요소는 대체로 문제의 분해, 공식화, 추상화, (프로그램을 통한) 자동화, 알고리즘, 최적화와 같이 큰 틀에서 문제해결 절차를 포함하며, 무한반복, 재귀, 디버깅 같이 컴퓨터에서 중요하게 사용되는 개념이 그 핵심적인 구성요소(Grover & Pea, 2013)로 언급된다. Weintrop et al.(2016)은 STEM 교육 맥락에서 수학·과학 교과에서 CT를 분류하였는데([그림 II-1]), 문제해결(모델링과 시뮬레이션)에 개념이해, 풀이와 검사를 목적으로 한 컴퓨터 활용을 CT에 포함시켰다.

본 연구에서는 선행 연구 고찰에 근거하여 CT를 컴퓨터 사용 여부와 상관없이 문제해결 관련 사고이며 문제해결을 위한 컴퓨터 사용을 로 보았다. 그러나 실질적으로 컴퓨터 활용이 제한적인 국내 상황에서, 수학교육에 관해서는 컴퓨팅 환경에서의 문제해결에 주목하여 논의를 전개하였다.

CT, 컴퓨팅, 수학교육 사이의 관계를 살펴보면, CT는 컴퓨팅 도구의 명령세트 등의 영향을 받으나 컴퓨터 소양과는 다르며 또한 컴퓨팅이 항상 CT를 필요로 하는 것은 아니다. 또 컴퓨팅이 수학교육에 도움이 될 수 있으나 때로는 방해가 될 수 있기 때문에 컴퓨터의 긍정적 영향이 부각될 수 있기 때문에 주의가 필요하다. 컴퓨팅 환경에서의 수학교육이 CT 이외의 것을 다룰 수 있으며, 컴퓨팅 환경이 아닌 수학 수업에서 CT를 개발 또는 사용할 수도 있다. 컴퓨팅 환경에서 활성화되는 CT는 현재 수학교육 범위를 벗어나 있으나 컴퓨팅 환경에서는 낮은 학년 수준에서도 용이하게 다룰 수 있는 중요한 개념이 있다. 예를 들면, 재귀적 사고와 같은 패턴의 무한반복(iteration)이 그러한 사례이다.

CT는 문제해결을 위한 사고로 수학적 문제해결의 한 요소이기도 하다. 복잡한 상황을 분해하고 자신이 다룰 만한 적당한 크기의 모듈로 분해하는 일, 문제와 동일한 구조를 갖는 모델을 찾는 일은 CT의 요소이자 문제해결의 필수적인 과정이다. CT를 컴퓨터 교과 뿐 아니라 K-12 여러 교과에서도 통합적으로 접근하려는 해외 연구 사례들이 이러한 입장에 서있다. 계산적 사고(CT)는 컴퓨터 교과의 전유물이 아닌 것이다.

IV 장에서 CT와 수학교과의 통합은 문제해결과 표현에 컴퓨팅 환경이 지원될 것을 전제로 논의하였다. 첫째, 코딩을 수학 수업에 통합시켜 다루는 것이 한 방법이다. 수학적 문제해결을 위한 프로그래밍은 수학적 이해와 분석적 사고를 돕는다. LOGO나 엑셀을 활용한 간단한 코딩으로 지필환경에서는 효율성 때문에 다루기 힘든 재귀적 사고나 알고리즘 학습에 유용하며 절차와 과정을 분해하여 생각할 수 있게 돕는다. 둘

째, 컴퓨팅 환경의 지원은 실질적으로 학생들의 수학적 문제해결을 도와 수학교과와 CT를 통합을 가능하게 한다. 또 컴퓨터 활용은 수학적 사고의 영역을 확장될 수 있다. 셋째, STEAM 교육의 일환으로 수학이 학생들이 관심을 가지는 사회, 물리, 환경 문제를 다루는 것에서 CT를 개발하고 활용하게 할 수 있다.

CT와 수학교육의 통합을 위하여 이를 지원하는 교육과정 개정이 선행되어야 한다. 컴퓨터는 수학교육 방법 뿐 아니라 수학 내용과 지도 계열에 영향을 끼친다(NCTM, 2000, etc.)는 것을 인식하고 내용과 지도계열 등 세심하게 수학교육과정 개편이 이루어져야 한다. 컴퓨팅 환경에서의 학생활동이 기존 교육과정의 범위를 넘지 않도록 제한하는 것은 학생들의 문제해결을 향한 자발적인 의지와 적극적 사고를 무력화시키며, 결과적으로 학생들에게 수학은 지루하고 낡은 경험으로 남게 될 것이기 때문이다.

"새 술은 새 부대에!" 컴퓨팅 도구를 활용한 문제해결은 4차 산업혁명 시대를 살아갈 학생들에게 필수적인 경험이며 CT는 그 핵심개념이다. CT는 수학 교과에서도 현재 중요하게 사용되는 유용하고 개발이 가능한 사고이다. 그러나 컴퓨팅 환경이 제공되는 수학교실은 CT의 개발과 학습, 그리고 CT의 실질적 사용과 함께 학생들에게 문제해결의 성공적 경험을 지원할 것이다.

참고문헌

고호경, 김주후, 오우상, 양윤정, 장주호. (2016) **수학 중심의 STEAM_세상을 읽는 수학 교재**. (성과물 번호. DD16090010). 한국과학창의재단. http://kms.kofac.re.kr/openapi/kofac.html

교육부. (1992). **중학교 교육과정**. 교육부 고시 제1992-11호.

교육부. (2011). **수학과 교육과정**. 별책8.

교육부. (2015a). **수학과 교육과정**. 별책8(제2015-74호).

교육부. (2015b). **개정 정보 교육과정**. 별책10_실과(기술가정)_정보과 교육과정(제2015-74호).

교육부. (2015c). **소프트웨어 교육 운영 지침**. 서울: 교육부.

교육부・미래부. (2015.7.21.) [보도자료] "초등학교에서 대학까지, 소프트웨어(SW) 교육 청사진 나왔다 !"「SW중심사회를 위한 인재양성 추진계획」 http://www.msip.go.kr/web/msipContents/contentsView.do?cateId=mssw311&artId=1270998

권정인, & 김재현. (2016). 초・중등 정보교육과정과 Computational Thinking 평가요소에 관한 연구. **인터넷정보학회지, 17**(2), 47-52.

김석전, 전용주, & 김태영. (2015). Track 4 교수학습방법 및 시스템: 중학교 정보교육과정 교수학습방법 제안 (소프트웨어 교육 운영 지침을 중심으로). **한국컴퓨터교육학회 학술발표대회논문집, 19**(2), 105-109.

김화경. (2015) 거북 마이크로월드와 컴퓨팅 사고력. **수학교육에서 공학적 도구. 2015 연보**. 대한수학교육학회. 355-367.

나귀수. (2000). 수학교육에서 컴퓨터 활용에 대한 소고. **학교수학, 2**(1), 97-110.

이영준, 백성혜, 신재홍, 유헌창, 정인기, 안상진, 최정원, 전성균.(2014). **초중등 단계 Computational Thinking 도입을 위한 기초 연구**. 한국과학창의재단 연구보고서(BD14060010)

이화영, & 장경윤. (2015). 인지적 도구로서의 사칙계산기 활용. **학교수학, 17**(2), 157-178.

장경윤, 황우형, & 이중권. (2001). 탐구형 기하 소프트웨어 (Geometer's Sketchpad) 의 활동 자료 개발과 그 효과에 관한 연구. **수학교육**

학연구, 11(1), 193-206.

전영국. “컴퓨터 프로그래밍과 창의성 발현 활동에 관한 질적 사례 연구: NetLogo 기반의 계산적 사고 중심으로.” **컴퓨터교육학회논문지** 18.3 (2015): 1-14.

최숙영. (2011). 컴퓨터과학 및 교육 2: 21 세기 소양과 계산적 사고 관점에서의 컴퓨터과학 교육 분석. **한국컴퓨터교육학회 학술발표대회논문집, 15**(2), 99-103.

Abelson, H. (1981). *Turtle geometry*. MIT Press.

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Educational Technology & Society, 19*(3), 47.

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.

BBC Bitesize. Introduction to computational thinking http://www.bbc.co.uk/education/guides/zp92mp3/revision (Retrieved 20 Jan. 2017).

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Annual American Educational Research Association meeting*, Vancouver, BC, Canada.

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

Cuny, J., Snyder, L., and Wing, J. (2010). Computational Thinking: A Definition. *Unpublished manuscript*.

Cuoco, A. (1990). *Investigations in algebra*. MIT Press.

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501.

Gadanidis, G. (2017) “Artificial intelligence, computational thinking, and mathematics education”, *The International Journal of Information and Learning Technology*, Vol. 34 Issue: 2, pp.133-139, https://doi.org/10.1108/IJILT-09-2016-0048

Grover, S., & Pea, R. (2013). Computational Thinking in K-12 *A Review of the State of the Field. Educational Researcher, 42*(1), 38-43.

Hoyles, C., & Noss, R. (1992). *Learning mathematics and log*o. MIT Press.

Hu, C. (2011, June). Computational thinking: what it might mean and what we might do about it. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (pp. 223-227). ACM.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads, 2*(1), 32-37.

Lu, J. J., & Fletcher, G. H. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin, 41*(1), 260-264. http://scholar.google.co.kr/scholar?start=10&q=computational+thinking+in+mathematics+education&hl=ko&as_sdt=0,5

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.

National Council of Teachers of Mathematics (Ed.). (2000). *Principles and standards for school mathematics* (Vol. 1). NCTM.

National Research Council. (2011). *Report of a workshop on the pedagogical aspects of computational thinking*. National Academies Press.

Papert, S. *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc., 1980.

Polya, G. (1968). *Mathematics and Plausible Reasoning: Patterns of plausible inference* (Vol. 2). Princeton University Press.

Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. *Journal of Computing Sciences in Colleges, 25*(5), 66-71.

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. http://link.springer.com/article/10.1007/s10639-012-9240-x

Sutherland, R., & Balacheff, N. (1999). Didactical complexity of computational environments for the learning of mathematics. *International Journal of Computers for Mathematical Learning, 4*(1), 1-26. doi: http://dx.doi.org/10.1023/A:1009882419704

Trouche, L. (2003). From artifact to instrument: mathematics teaching mediated by symbolic calculators. *Interacting with Computers, 15*(6), 783-800.

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.

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 of London A: mathematical, physical and engineering sciences*, 366(1881), 3717-3725.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends, 60*(6), 565-568. doi: http://dx.doi.org/10.1007/s11528-016-0087-7

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* (TOCE), 14(1), 5.

A Feasibility Study on Integrating Computational Thinking into School Mathematics

Chang, Kyung Yoon (Konkuk University)

The purpose of this study was to gain insights into investigating the feasibility on integrating computational thinking(CT) into school mathematics. Definitions and the components of CT were varied among studies. In this study, CT in mathematics was focused on thinking related with mathematical problem solving under ICT supportive environment where computing tools are available to students to solve problems and verify their answers. The focus is not given on the computing environment itself but on CT in mathematics education. For integrating CT into mathematical problem solving, providing computing environment, understanding of tools and supportive curriculum revisions for integration are essential. Coding with language specially developed for mathematics education such as LOGO, and solving realistic mathematical problems using S/W such as Excel in mathematics classrooms, or integrating CT into math under STEAM contexts are suggested for integration CT into math education. Several conditions for the integration were discussed in this paper.

* Key Words : computational thinking(계산적 사고, 컴퓨팅 사고), LOGO(LOGO) Excel(엑셀), integration(통합), computing environment(컴퓨팅 환경), programming(프로그래밍), Recursion(재귀), representation(표현) STEAM(융합교육)

논문접수 : 2017. 8. 13
논문수정 : 2017. 9. 3
심사완료 : 2017. 9. 7