

유아용 코딩아트 교육을 위한 커리큘럼 설계*

안성혜**(상명대학교)

국문초록

현대사회는 소프트웨어(software)를 중심으로 한 융합기술사회로 변화되고 있으며, 4차 산업혁명 시대에 필요한 창의융합인재를 육성하기 위해 컴퓨팅 사고력(Computational Thinking, CT)의 중요성이 강조되고 있다. 본 연구는 처음으로 코딩(coding) 교육을 접하게 되는 유아(5-7세) 및 초등 저학년(8-9세)을 대상으로 컴퓨팅 사고력을 배양하기 위한 코딩교육의 커리큘럼을 연구하고자 하는 목적을 가진다. 먼저, 유아 및 초등저학년의 연령별 특성을 반영한 코딩교육의 방향성을 설정하고자 했으며, 이를 위해 소프트웨어교육의 교수학습 모형 분석과 유아용 코딩교육 콘텐츠의 유형별 사례분석을 통해 그 교육적 효과를 살펴보았다. 그 결과, 유아용 코딩교육은 언플러그드 형태의 STEAM 융합교육과 창의적인 문제해결 중심의 프로젝트 교육이 효과적이라는 연구결과가 많았다. 따라서 본 논문에서는 수학, 과학, 기술, 예술 활동을 융합한 STEAM 교육이자, 프로젝트 중심의 언플러그드 코딩아트(coding art) 교육 모델을 제안하고자 한다. 즉, 유아 대상의 코딩교육 교육과정은 컴퓨팅 사고력과 디자인 사고력(Design Thinking, DT)을 키우기 위한 교육목표를 가지며, 이를 바탕으로 컴퓨터 과학의 개념과 컴퓨팅 사고력, 디자인 사고력을 키우기 위한 총 25개의 학습내용을 구성하였다. 또한 유아들이 재미있고 능동적으로 학습할 수 있도록 보드게임을 활용한 개념이해 활동과 학습내용을 창의적인 작품으로 표현하는 프로젝트 기반의 미술활동으로 교육과정을 설계하였다.

주요어 : 코딩교육, 코딩아트, 컴퓨팅 사고, 디자인 사고, STEAM 융합교육, 프로젝트 학습

I. 서론

현대사회는 소프트웨어(software)를 중심으로 한 융합기술사회로 변화되고 있으며, 인공지능(AI), 사물인터넷(IoT), 빅데이터(Big Data) 분석 및 활용, 자율주행 자동차, 스마트 공장(Smart Factory), 가상/증강현실(VR/AR) 등 미래 유망 기술이 화두로 떠오르고 있는 4차 산업혁명 시대에 필요한 창의 융합인재를 육성하기 위해 컴퓨팅 사고력(Computational Thinking, CT)의 중요성이 강조되고 있다. 카네기멜론 대학의 교수인 Jennette M. Wing(2006)은 컴퓨팅 사고력이 모든 사람을 위한 기반 기술이며, 컴퓨터공학의 기본 개념과 기술, 도구를 활용하여 현실의 복잡한 문제를 해결하는 시스템을 설계하고, 인간의 행동을 이해할 수 있는 사고방식(이현민, 2018)이라고 정의하고 있다. 이러한 컴퓨팅 사고력을 키워주기 위한 기초교육으로 코딩(coding) 교육이 주목받고 있다. 빌 게이츠는 코딩교

* 본 논문은 2017년도 상명대학교 교내학술연구비 지원을 받아 연구되었음.

** 정교수, ramsuny@smu.ac.kr

육이 “사고의 범위를 확장하고 문제 해결력을 키워준다”고 했고, 스티브 잡스도 “생각하는 방법을 배우기 위하여 모든 사람은 코딩교육을 받아야 한다”고 역설했다. 코딩 교육은 문제를 분석하고 유형화하는 분석 작업과 알고리즘을 작성하는 과정을 통해 논리적 사고와 문제해결 역량, 창의력 등을 키울 수 있기 때문이다(송상수, 2015).

이스라엘은 세계에서 가장 빠르게 1994년에 소프트웨어 과목을 정규교육 과정에 포함시켰고, 일본은 2009년에 소프트웨어 교육을 필수과목으로 지정하였으며, 인도는 2010년부터 SW교육을 초·중·고에서 의무화하였고, 영국에서는 2014년부터 코딩교육을 초·중등학교 필수교과로 지정하면서 본격적인 코딩 교육 붐이 일어나게 되었다. 국내 역시 2015년 개정 교육과정을 통해 문, 이과 통합 및 소프트웨어 교육을 초·중·고 교과과정에 필수과목으로 지정하였으며, 컴퓨팅 사고력 함양이란 목적 하에 컴퓨터 과학의 세부교육으로 집중하고 있다. 2018년부터는 중·고등과정에, 2019년부터는 초등과정에 코딩 교육이 필수과목으로 시행됨에 따라 소프트웨어 창의 인재 양성을 위한 다양한 정부시책 및 지원이 늘어나고 있으며, 유치원을 비롯하여 초·중등학교 대상의 코딩 교육과 관련된 온·오프라인 사설 교육기관 및 교육콘텐츠 산업이 빠르게 성장하고 있는 추세이다.



그림 1. 국내 코딩교육의 교육 목표

출처 : 교육부 소프트웨어 교육 선도 교원연수 교재(2016)

여기서 코딩이란 ‘컴퓨터 프로그래밍 작업’으로 정의할 수 있으며, 용어를 좀 더 넓게 해석하자면, ‘프로그램 언어를 이용해 시스템을 다루는 작업’으로도 이해할 수 있다(사이언스타임즈, 2019). 그래서 학교수업은 물론, 사교육 시장에서도 코딩 교육을 소프트웨어 교육과 동일하게 생각하여 접근하는 곳이 많으며, 또한 소프트웨어 활용법의 단순 암기식 코딩 교육이 양산되고 있는 형국이다. 그러나 코딩을 처음으로 접하게 되는 유아 및 초등 저학년 어린이를 대상으로 하는 코딩 교육은 단순히 소프트웨어의 활용 능력만을 키워주는 것이 아니라 생각하는 방법과 문제해결능력, 추론과 논리적 사고력 등의 더 큰 학습능력을 갖추기 위한 토대 교육이 반드시 필요하다. 코딩 교육의 궁극적인 목적은 실생활에 필요한 문제 해결을 위한 논리적 사고력을 키우며, 문제해결 과정을 통해 스스로 창의적인 생각을 하도록 하는 것을 의미하기 때문이다(김성환, 2017). 따라서 유아를 위한 코딩 교육은 유아의 눈높이에 맞추어 절차적 사고를 통한 알고리즘 원리를 이해시킬 수 있어야 하며, 창의적인 사고력을 키우기 위한 놀이와 게임 등으로 자연스럽게 코딩을 이해할 수 있도록 교육과정이 설계되어야 한다.

현재 유아나 초등저학년 어린이를 대상으로 개발되어 있는 코딩교육 콘텐츠의 사례들을 살펴보면, 컴퓨터 프로그래밍 용어는 사용하지 않으면서 반복문 또는 연산 원리를 알려주는 언플러그드 코딩 보드게임의 형태와 캐릭터나 로봇을 활용한 길 찾기 게임형식의 피지컬 컴퓨팅 도구, 스크래치나 엔트

리와 같은 코딩 교육 소프트웨어를 활용한 블록 코딩 개념이 대부분이다. 이는 실제 교육현장에서 코딩 교육을 진행할 강사가 매우 부족한 상황에서 쉽게 코딩 교육을 지도할 수 있는 대안으로 마련된 것이라 판단된다. 게다가 이제 막 학교 정규 교과과정에 코딩 교육이 도입되기 시작한 단계로 다양하고 특화된 교육보다는 보편적인 교육과정의 보급이 더 시급한 상황이므로 컴퓨팅 사고력을 키우기 위한 창의적인 문제해결형의 커리큘럼이 드러난 교육사례를 찾아보기는 매우 드문 상황이다.

또한 유아나 초등 저학년을 대상으로 하는 코딩 교육은 즐겁게 놀이를 통해서 코딩에 대한 부담감을 없애고 자연스럽게 컴퓨팅 사고력을 기르게 할 수 있다는 장점(경남신문, 2017)이 있음에도 불구하고 우리나라 대부분의 코딩 교육과 관련된 연구는 초등학교 이상에서 이루어지고 있으며, 유아를 대상으로 한 연구는 상당히 부족한 실정이다(이애화, 2018).

이에 본 논문은 처음으로 코딩 교육을 접하게 되는 유아(5-7세) 및 초등 저학년(8-9세)을 대상으로 컴퓨팅 사고력을 배양하기 위한 언플러그드 코딩 교육의 커리큘럼을 연구해보고자 하는 목적을 가진다. 이를 위해 유아 및 초등 저학년의 특성을 반영하여 STEAM 융합교육으로서의 컴퓨팅 사고력과 논리, 수학, 과학, 미술 활동을 연계한 프로젝트 중심의 코딩아트(coding art) 교육모델을 제안하고자 한다. STEAM 융합교육은 다양한 정보를 주어진 문제에 따라 융복합적인 사고로 해결할 수 있는 창의적 인재를 양성하는 데에 목표를 두고 있으며, 미래의 과학기술 분야 인재에게 지식뿐만 아니라 상상력, 인간의 감성까지 아우를 수 있는 균형 감각을 제공할 수 있기 때문이다(조향숙, 김훈, 허준영, 2012).

II. 유아용 코딩교육 콘텐츠의 분류

1. 코딩 교육용 로봇

유아용 코딩로봇 교구들은 완성된 형태의 로봇과 스마트 블록 등으로 제공되어 지며, 만3세 이상의 유아 및 미취학 아동을 위한 제품으로 많이 출시되고 있다. <비봇(bee-bot)/블루봇(blue-bot)>, 구글 블로리 기반의 <오조봇(ozobot)>, <알버트>, <코딩애벌레>, <코딩마우스> 등이 있다. <비봇>은 영국 유치원에서 사용하는 코딩교육용 로봇으로 7개의 쉬운 명령어를 이용한 간단한 조작을 통해 코딩을 배울 수 있다. 등에 위치한 명령어를 입력하는 횟수로 비봇이 움직이게 되며, 목적지에 도달하기 위한 목표 설정을 통해 수학적 문제해결력과 연관이 있음을 알 수 있다. 이 과정에서 유아들은 문제 해결을 위한 가장 기본적인 알고리즘 사고 및 논리적 사고를 할 수 있게 된다(이연승, 성현주, 2017). 박성현(2002)과 이연승 및 성현주(2017)는 비봇을 활용한 수학적 문제해결력 증진 프로그램을 통해 유아의 수학적 문제해결능력이 향상되는 연구결과를 도출했으며, 이는 유아의 실행기능, 전환, 작업 기억이 유아의 수학적 문제해결력에 모두 영향을 미친다는 이경진, 이지현(2017)의 연구결과와 맥락을 같이한다. 또한 <비봇>의 경로를 찾기 위해 탐구하고 <비봇>을 수와 관련하여 조작해 봄으로써 탐구중심 수·조작영역 활동이 유아의 수학적 문제해결력 및 태도에 영향을 미친다는 이명옥(2012)의 연구와도 상통한다.



그림 2. 코딩용 로봇의 사례_비봇(좌)과 오조봇(우)

2. 코딩 교육용 소프트웨어

유아 대상의 코딩 교육용 소프트웨어는 블록형 비주얼 코딩언어가 대부분이며, 마우스로 드래그하면서 명령어를 끌어오는 방식으로 학습자 중심의 체형형 프로그래밍 교육 언어(김수환, 임선영, 송상수, 2015)이다. 미국의 <스크래치(Scratch)>, 한국의 <엔트리(Entry)>, 일본의 <프로그래밍(プログラミング)>, 구글의 <블럭키(Blockly)>, 코드 스팩(CodeSpark)의 <The Foo s>(http://thefoos.com/) 등이 블록형 소프트웨어의 유형이다.

<스크래치>는 2006년 미국 매사추세츠공과대(MIT)에서 개발한 아동용 코딩교육 소프트웨어로 플래시 기반으로 만들어져서 이미지 효과, 미디어 처리 기능이 뛰어난 장점이 있으며, 명령어가 적힌 블록을 끼워 맞추며 놀이하듯 코딩을 배울 수 있도록 설계되었다. 학습이 쉽고 교육적 확장이 가능한 장점을 가진 스크래치는 현재 40여개 언어 150여개 국가의 초등학교에서 대학까지 SW교육을 위해 사용되고 있다.

<엔트리>는 2013년 한국에서 만들어진 코딩 교육용 소프트웨어로 자바스크립트 기반으로 제작되어 호환성이 뛰어나기 때문에 모바일, 태블릿PC에서도 별도의 앱 없이 사용할 수 있다. <엔트리>와 <스크래치>는 모두 창의력이 요구되는 코딩교육 프로그램으로 학습자가 작은 명령 단위인 블록 조각을 서로 조립하여 자신의 의사대로 캐릭터의 행동을 지정하거나 환경을 설정하고, 이야기나 게임 구성을 수행하여 인터랙티브한 이야기, 게임, 애니메이션을 직접 만들 수 있다. 또한, 코드보기(스크립트보기) 기능을 제공하고 있어서 온라인 커뮤니티에서 자신의 작품을 다른 사람들과 공유하면서 창의적 사고, 체계적 추론, 협동 작업의 능력을 배울 수 있도록 되어있다(이현민, 2018).

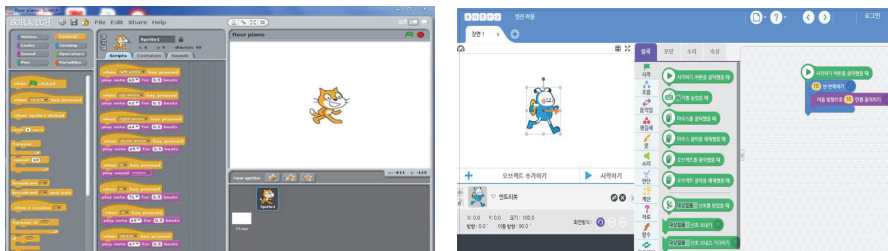


그림 3. 코딩용 소프트웨어의 사례_스크래치(좌)와 엔트리(우)

3. 코딩 교육용 보드게임

코딩 교육용 보드게임은 컴퓨터 없이 보드게임만으로 컴퓨터 과학의 개념과 알고리즘, 프로그래밍의

기본 개념을 이해시킬 수 있는 언플러그드 코딩교육 도구이다. 대개 6~8세 정도면 충분히 개념을 이해할 수 있는 수준으로 제작되었으며, 1인용에서 6인용까지 있어서 팀으로 함께 교육할 수도 있다. 컴퓨터과학의 개념을 이해할 수 있는 <요원12>, <시그널>, <go to stop>, <ZIP>과 알고리즘, 순차, 반복, 조건 등의 프로그래밍 개념을 이해할 수 있는 <코드마스터>, <엔트리봇>, <프로봇>, <플레이 로봇>, <마이크로봇>, <맛있는 코딩>, <스택버거> 등이 있으며, 스크래치와 엔트리를 보드게임으로 이해할 수 있는 <catch the cat>, <catch the dog>가 있다. <코드마스터(code master)>는 60레벨의 맵으로 구성된 프로그래밍 논리게임으로 순차적으로 어떤 토큰을 사용해야 하는지 고민하는 과정에서 알고리즘의 이해와 절차 사고력이 증진되며, 게임에 주어진 조건을 분석하는 과정에서 자료 분석력을 기를 수 있다.



그림 4. 코딩용 보드게임의 사례_코드마스터(code master)

한선관과 신수범(2011)은 구체적인 사례와 게임, 퍼즐과 놀이 활동을 바탕으로 한 언플러그드 에듀테인먼트 프로그램 연구를 통해 학생들의 수업 참여도와 학습 성취도를 높일 수 있다는 결과를 도출하였으며, 언플러그드 에듀테인먼트 수업모형을 문제해결모형, 발견학습모형, 담구학습모형의 3가지로 제시하였다. 놀이학습 기반의 언플러그드 교육이 수학적 인지능력 발달(김형식, 2015) 및 창의성과 문제해결 능력에 효과적(구영은, 2015)이며, 정민경(2017)은 언플러그드 컴퓨팅을 활용한 STEAM 활동이 유아의 창의성과 문제해결력에 긍정적인 영향을 준다고 하였다.

4. 코딩 학습 프로그램 세트

그밖에도 피지컬 교구와 디지털 플레이 앱, 메이커 놀이 활동이 결합된 단계별 코딩학습 프로그램인 <큐비코(cubico)>가 있다. <큐비코(cubico)>는 아동 발달 과정에 맞도록 설계된 보드(스테이지 보드 위로 코딩칩을 배치하는 프리코딩), 블록(모듈 위로 직접 블록을 순서대로 삽입하여 코드를 짜는 실전), 앱 실행(코딩 결과를 디바이스의 게임 애니메이션을 통해 확인)의 3단계 학습으로 구성되는 총 58개의 스테이지를 체험할 수 있어 아이들의 흥미를 이끌 수 있는 놀이형 프로그래밍 교육 도구이다. 최근 유아용 코딩교육 콘텐츠는 언플러그드 놀이 활동과 피지컬 교구 및 로봇, 디지털 앱, 프로젝트 기반 메이커 활동에 이르기까지 모두 하나로 융합된 교육 프로그램의 형태로 진화하고 있다.



그림 5. 코딩학습 세트 <큐비코(cubico)>

Ⅲ. 유아용 코딩교육의 방향성

유아를 대상으로 하는 코딩교육은 단순히 소프트웨어를 활용한 컴퓨터 프로그래밍 교육이 아니라 생각하는 방법과 창의적인 문제해결능력, 추론과 논리적 사고력 등을 키워주는 토대교육이 바탕이 되어야 한다. 또한 유아의 눈높이에 맞게 절차적 사고를 통한 알고리즘의 원리를 이해시킬 수 있어야 하며, 창의적인 사고력을 키우기 위한 놀이와 게임으로 자연스럽게 코딩을 이해할 수 있도록 교육과정이 설계되는 것이 바람직하다. 따라서 본 연구에서는 유아용 코딩교육의 방향성을 크게 4가지로 요약하여 제안하고자 한다. 즉, 컴퓨팅 사고력과 디자인 사고력을 키울 수 있으면서, 프로젝트 기반의 창의적인 문제해결학습이 가능하도록 교육내용을 구성하고, 유아의 발달단계에 적합한 단계별 교육과정을 설계하는 것이다.

1. 컴퓨팅 사고력(Computational Thinking, CT) 교육

최근 전 세계적으로 코딩 열풍을 불러일으키고 있는 영국, 핀란드, 호주 등에서는 유치원부터 코딩교육을 의무적으로 실시하고 있다. 국내에서도 코딩 교육을 실시하고 있는 유치원이 늘고 있으며, 기업들은 유아용 코딩 교육 콘텐츠 및 교구를 활발하게 개발하고 있다. 컴퓨터나 스마트폰 사용이 보편화된 요즘 유아들에게 있어 코딩 교육은 쉽게 받아들여질 수 있으며, 이제는 ‘게임하는 아이’가 아니라 오히려 ‘게임 만드는 아이’가 되도록 독려할 수 있는 환경이기도 하다. 코딩 교육의 목적인 컴퓨팅 사고, 문제해결능력에 대한 역량의 강화는 자율성과 창의성을 기르는 데 중점을 두고, 전인발달을 이루도록 구성한다는 3-5세 연령별 누리과정의 주요 구성방향의 내용과 연결된다(이연승, 최현주, 2017).

유아용 코딩교육의 핵심은 컴퓨터 프로그래밍을 배우는 것이 아니라 컴퓨팅 언어들을 문제에 알맞게 잘 사용하는 창의성과 사고능력을 키워주는 데 있다. 어떠한 문제를 마주했을 때 그 문제를 잘게 쪼개고, 분석하고, 패턴을 읽고, 해결하기 위한 방법을 구조화(알고리즘 설계)하는 등의 과정을 통해 아이들은 같은 문제를 다양한 방법으로 해결해 낼 수 있는 능력을 갖는다. 이 과정에서 습득하는 것이 컴퓨팅 사고력이다. 컴퓨터 사고력이란 컴퓨터 과학의 기본 개념과 원리 및 컴퓨팅 시스템을 활용하여 실생활 및 다양한 학문분야의 문제를 이해하고 창의적 해법을 구현하여 적용할 수 있는 능력이다(교육부, 2015).

Wing(2006)에 의하면 컴퓨팅 사고력의 핵심요소는 크게 추상화(abstraction)와 자동화(automation)로 나뉜다. 추상화는 실제 세계의 문제를 해결 가능한 형태로 표현하기 위한 사고 과정으로, 문제를 해결하기 위해 세부적인 것을 제거하여 단순화하는 과정이며, 공통적인 핵심과 본질을 찾아 일반화하는 과정(정인기, 2016)이다. 자동화는 추상화 과정에서 만들어진 해결 모델을 컴퓨팅 시스템을 통해 효과적으로 수행되도록 표현하여 작동시키는 것이다(한선관, 2017). CSTA & ISTE(2011)에서는 컴퓨팅 사고력을 자료수집, 자료분석, 자료표현, 문제분해, 추상화, 알고리즘과 절차, 자동화, 시뮬레이션, 병렬화 등으로 세분화하고 있으며, 각각에 대한 설명은 <표 1>과 같다.

표 1. 컴퓨팅 사고력의 개념과 설명

Wing	CSTA & ISTE	설명
추상화 (광의)	자료 수집	해결해야 하는 문제와 관련된 알맞은 자료를 모으는 과정
	자료 분석	자료를 이해하고, 패턴을 찾아 결론을 도출하는 과정
	자료 표현	적절한 그래프, 차트, 글, 그림 등으로 자료 정리하고 표현하는 과정
	문제 분해	문제를 해결가능한 수준의 작은 문제로 나누는 과정
	추상화(협의)	문제해결을 위해 필요한 핵심 요소를 파악하고, 복잡함을 단순화시키는 과정
자동화	알고리즘 및 절차	문제해결이나 목표를 달성하기 위해 수행하는 일련의 순차적인 과정
	자동화	컴퓨팅 시스템으로 반복 수행할 수 있는 형태로 패턴화하는 과정
	시뮬레이션	문제해결을 위해 만든 모델을 실행시켜 결과를 파악하는 과정
	병렬화	목표달성을 위해 독립적인 작업을 동시에 수행하도록 자원을 구성하는 과정

출처 : ISTE & CSTA(2011), 박성립(2016), 오경선, 안성진(2016)

2. 디자인 사고력(Design Thinking; DT) 교육

유아 코딩교육의 목적은 아이들이 스스로 생각하고 문제를 해결할 수 있는 다양한 방법을 찾는 과정을 배우는 것으로, 이를 통해 논리적 사고력, 창의적 사고력, 표현력, 문제해결력을 키울 수 있게 된다. 즉, 컴퓨터를 조작하여 나오는 결과물보다는 그 결과를 어떤 방법으로 도출했는가에 대한 사고 과정을 중시하여 컴퓨터처럼 생각하는 능력을 기르는 것이다. 창의적인 문제해결 과정은 문제를 발견하고, 분석하며, 분류하여 표현하고, 문제를 해결하기 위한 전략을 수행하며, 문제해결의 창의적인 아이디어를 도출하고, 이를 구체적으로 시각화하여 실행하는 일련의 과정으로 구성되며, 이 과정에서 디자인 사고력(Design Thinking)이 필요하게 된다.

이지선(2015)은 컴퓨터 사고를 기반으로 한 컴퓨터 교육에 디자인적 사고를 적용한 연구에서 디자인 사고의 프로세스를 따라 체계적인 프로젝트 개발을 한 경우 창의적 결과물이 더 많이 도출되었고, 자신의 아이디어로 만들면서 학습하는 경우 몰입도와 학습 능력 향상에 긍정적 영향이 있다는 결과를 발표했다(이승철, 2018). 디자인 사고는 디자인 분야에서 시작된 문제해결 프로세스와 사고 방법으로, 문제에 대한 공감(empathy)과 생각의 시각화(visualization)를 강조하는 창의적인 발상 모델이며, 분석적 사고와 직관적 사고의 균형을 이루는 새로운 사고방식이다(Carroll, M. et al., 2010; Brown, T., 2008; Martin, R., 2009; 이도현, 2015).

디자인 사고의 프로세스(design thinking process)는 디자인 사고를 행동으로 구현하는 것으로서 아이디어를 생성하고 발전시켜 나가는 문제해결 접근법이다(김진옥, 2018). 디자인 사고의 프로세스는 공감(Empathize)-문제정의(Define)-발상(Ideate)-프로토타입(Prototype)-적용(Test)의 관점(김자민, 2015)과 발견하기-해석하기-아이디어내기-실험하기-발전시키기로 보는 견해(최정아, 2016;

IDEO, 2011) 등이 있다. 본 논문에서는 코딩교육을 위한 컴퓨팅 사고에 대응하는 개념으로 디자인 사고 프로세스에 접근하고자 하며, 문제 발견-문제 분석-문제 정의-문제해결 전략-아이디어 발상-설계-프로토타입-시연 및 평가로 정의하고자 한다. 디자인 사고력은 다시 논리적 사고, 분석적 사고, 추상적 사고, 전략적 사고, 확산적 사고, 절차적 사고, 수렴적 사고, 비판적 사고 등으로 세분화될 수 있으며, 어릴 적부터 이러한 사고가 잘 발달할 수 있도록 교육하는 것이 필요하다.

표 2. 컴퓨팅 사고에 대응하는 디자인 사고력의 구성요소

컴퓨팅 사고(CT)		디자인 사고(DT)		내용
추상화 능력	자료 수집	문제 발견	논리적 사고	탐색, 이해, 발견, 자료
	자료 분석	문제 분석	분석적 사고	분류, 규칙, 패턴, 정보
	자료 표현	문제 정의	추상적 사고	단순화, 시각화, 추상화
	문제 분해	문제해결 전략	전략적 사고	분해, 조건, 추론, 압축
	추상화	아이디어 발상	확산적 사고	연상, 유추, 조합, 관계
	알고리즘과 절차	설계	절차적 사고	구조, 절차, 연산, 협동
자동화 능력	시뮬레이션	프로토타입	수렴적 사고	실행, 함수, 변수, 이벤트
	병렬화	시연 및 평가	비판적 사고	변별, 선택, 반복, 감상

출처 : 이은경(2009), 하희정(2017), 윤혜진(2018)에서 편집

3. 발달심리학적 구성주의 교육

이현진(2018)은 발달심리학에 근거하여 코딩교육 관련 커리큘럼을 연구한 선행연구들로부터 어린 학습자의 인지 발달 수준에 맞추어 단계별로 구분된 교육과정이 필요하다고 제안하였으며, 만 4-5세 이상의 어린이를 대상으로 코딩교육 시작이 가능하다고 밝혔다. 본 논문에서는 미국과 영국, 한국에서 코딩교육의 연령별 또는 단계별 교육과정이 어떻게 구성되고 있는 지 알아보았다.

CSTA(2011)는 미국의 학제인 K-12에 따라 컴퓨터과학 교육 분야를 컴퓨팅 사고(Computational Thinking), 협동(Collaboration), 컴퓨팅 실습 및 프로그래밍(Computing Practice & Programming), 컴퓨터 및 통신 장치(Computers and Communications Devices), 지역 사회의 글로벌 및 윤리적 영향(Community Global, and Ethical Impacts)의 5개 영역으로 구분하고, 교육과정을 <표 3>과 같이 3단계로 구분하고 있다. 각각의 단계를 살펴보면, 레벨1에서는 컴퓨터과학의 기본 개념에 대한 이해와 창작과 탐구에 초점을 맞추어 시퀀싱 위주로 배우고, 레벨2에서는 문제해결 도구로써 컴퓨팅 사고를 활용하여 의사소통과 협력적인 프로그래밍을 경험하며, 레벨3에서는 실제 제품개발의 프로젝트 관리를 통해 문제해결력을 배우게 된다.

표 3. 미국 K-12 컴퓨터 과학의 교육과정

레벨	나이	교육 내용
Level 1	grades K-6	컴퓨터 과학의 기본적인 개념, 창작과 탐구 활동, 시퀀싱(sequencing)
Level 2	grades 6-9	컴퓨팅 사고, 문제해결, 협력적인 프로그래밍(collaborative programming) 알고리즘을 통한 문제해결력(algorithmic problem solving), 실제 제품개발, 협력 학습, 프로젝트 관리, 의사소통
Level 3	grades 9-12	grades 9-10 Computer Science in the Modern World
		grades 10-11 Computer Science, Concepts and Practices
		grades 11-12 Topic in Computer Science

출처 : Computer Science Teachers Association(CSTA, <http://csta.acm.org/>, 2011), 이현진(2018)에서 편집

〈표 4〉는 영국의 컴퓨팅 교과과정의 연령별 교육과정을 정리한 것으로 크게 4단계로 구분하고 있는데, stage 1에서 알고리즘의 이해를 통해 간단한 프로그램을 체험해보고, stage 2에서는 프로그램 설계와 디버깅을 경험하면서 순차, 반복, 변수 등의 프로그래밍 언어를 배우게 되며, stage 3에서는 실제 세계 문제를 모델화하는 추상화를 통해 창의적 프로젝트를 수행하게 되며, stage 4에서는 문제해결을 위한 CT 기술을 계발하고 적용하는 단계로 구성된다.

표 4. 영국 컴퓨팅 교과과정의 시기별 주요내용

레벨	나이	교육 내용
Key Stage1	만 5-7세	알고리즘의 이해, 간단한 프로그램 제작 및 디버깅, 논리적 추론
Key Stage2	만 7-11세	프로그램 설계, 디버깅, 문제분해, 순차, 반복, 선택 및 변수, 입출력 사용, 알고리즘의 동작원리, 네트워크의 이해, 검색활용
Key Stage3	만 11~14세	실세계 문제를 모델화하는 컴퓨팅적 추상을 설계, 활용 및 평가 CT를 반영한 주요 알고리즘의 이해(검색, 정렬 알고리즘, 조건문) 도전적인 목표를 달성하기 위한 창의적 프로젝트 수행
Key Stage4	만 14~16세	컴퓨터과학, 미디어, 정보기술 분야의 지식, 창의성, 능력 함양 분석적, 문제해결, 설계, CT 기술을 계발하고 적용하기

출처 : 영국교육부(2013), 전용주(2017)에서 편집

한국의 코딩교육 교과과정은 2015년 개정된 교육과정에서 제시하는 초·중·고등학교의 교과과정을 토대로 각 단계별 교육내용과 프로그래밍 언어의 유형을 정리하였으며, 5-7세 유아와 초등 저학년의 교과과정을 추가하여 〈표 5〉로 제시하였다. 5-7세의 유치원생은 언플러그드 형태의 조작과 놀이로 컴퓨팅 사고를 이해하도록 하는 것이 바람직하며, 초등 1-2학년생은 언플러그드와 블록 형태로 프로그래밍을 체험하면서 알고리즘과 컴퓨팅 사고를 배우는 것이 효과적이다.

표 5. 한국의 코딩교육 교과과정

레벨	나이	교육용 언어	교육 내용
유치원	5-7세	unplugged	구체적 조작 활동 형태의 놀이(순차, 선택), 컴퓨팅 사고 이해
초등학교	1-2학년	unplugged or block	알고리즘과 프로그래밍 체험, 컴퓨팅 사고 서술적 표현(자료수집, 자료분석, 자료표현, 문제분해)
	3-6학년	block	문제이해와 해결방법 탐색, CT(변수, 연산, 반복문, 조건문) 순서도 표현(자료표현, 문제분해, 도식화, 추상화, 알고리즘) 컴퓨터언어(EPL, 코딩, 자동화, 병렬화)
중학교	1-3학년	visual & text	간단한 알고리즘 설계, 프로그램 개발, 컴퓨팅 사고로 문제 해결 CT(추상화, 알고리즘, 자동화, 병렬화)
고등학교	1-3학년	text	효율적 알고리즘 설계, 프로그램 개발, 창의적 융합문제 해결 CT(시뮬레이션, 디버깅, 비교분석, 창의적표현)

출처 : 교육부(2015), 하희정(2017)에서 편집

4. 프로젝트 기반의 창의적인 문제해결학습 교육

소프트웨어 교육의 교수학습모델은 행동주의, 인지주의, 구성주의적 관점을 고려하여 학습목표인 지식, 기능, 태도 영역이 포함되도록 개발되었으며(한국교육개발원, 한국교육학술정보원 2015), 특히 소

프트웨어 교육의 교수학습 모형은 컴퓨팅 사고 신장이라는 목표를 달성하기 위해 각 단계별 활동 내에 컴퓨팅 사고의 구성요소인 분해, 패턴인식, 추상화, 알고리즘, 프로그래밍을 포함하도록 설정하고 있다.

또한 구성주의 학습이론을 기반으로 하는 대표적인 학습 이론으로 프로젝트 기반 학습(project-based learning), 문제 중심 학습(problem-based learning), 창의적 문제해결학습(creative problem solving) 등 다양한 형태의 교수학습 방법들이 활용되고 있다. 전용주(2017)는 창의적 문제해결 학습을 코딩 교육에 적용하는 새로운 교육과정을 개발하고, 검증하는 연구를 진행하였다. 그 결과 초등학생 및 중학생을 대상으로 한 창의적 문제해결(CT-CPS) 수업모형 기반의 소프트웨어 수업이 창의적 문제해결력, 메타인지, 컴퓨터 학습태도, 학습동기 등 인지적, 정의적 영역에서 효과가 있음을 입증하였다. 따라서 본 논문에서는 <표 6>과 같이 소프트웨어 교육의 교수학습모형을 새롭게 6개로 정리하였다.

표 6. 소프트웨어 교육의 교수학습 모형

수업 모형	수업 절차	설명
시연중심 모델	시연	교사의 설명과 시범, 표준 모델 제시
	모방	학생 모방하기, 질문과 대답
	제작	단계적, 독립적 연습, 반복활동을 통한 기능습득
재구성중심 모델	놀이	학습자 체험활동, 관찰과 탐색
	수정	교사가 의도적으로 모듈, 알고리즘을 변형하여 제시
	재구성	놀이와 수정활동을 확장하여 자신만의 프로그램을 설계/제작
개발중심 모델	탐구	탐색과 발견을 통한 지식 구성
	설계	알고리즘의 계획 및 설계
	개발	프로그래밍 언어로 구현 및 피드백
디자인중심 모델	요구분석	주어진 문제에 대한 고찰과 사용자 중심의 요구분석
	디자인	분해와 패턴찾기, 알고리즘의 설계
	구현	프로그래밍과 피지컬컴퓨팅으로 산출물 구현
	공유	산출물 공유와 피드백을 통한 자기성찰
CT요소중심 모델	분해	컴퓨터가 해결 가능한 단위로 문제 분해
	패턴인식	반복되는 일정한 경향 및 규칙의 탐색
	추상화	문제 단순화, 패턴인식으로 발견한 원리 공식화
	알고리즘	추상화된 핵심 원리를 절차적으로 구성
	프로그래밍	컴퓨터가 이해할 수 있는 언어로 구현/실행
CT-CPS중심 모델	문제인식 및 분석	문제 발견, 관련 자료 수집 및 분석, 분류, 시각화로 제시
	아이디어 구상	문제 해결 방법, 핵심요소 찾기, 단순화, 추상화
	설계	문제해결 아이디어의 설계, 논리적인 순서도, 스토리보드 작성
	구현 및 적용	해결책의 구현, 프로토타입 제작, 발표 및 시연, 공유

출처 : 한국교육개발원, 한국교육학술정보원(2015), 전용주(2017)

IV. 유아용 코딩 아트(coding art) 교육의 커리큘럼

본 논문에서는 앞서 제시한 교육방향에 따라 컴퓨팅 사고력과 디자인 사고력 배양을 교육목표로 하는 유아용 코딩교육의 커리큘럼을 설계하고자 한다. 앞서 선행연구들을 분석한 결과, 유아를 대상으로 하는 코딩 교육과정은 STEAM 융합교육 방식으로 설계하는 것이 바람직하다고 생각하는데, 그 중

에서도 ART 프로젝트를 중심으로 하는 코딩교육 적용을 제안하려고 한다. 이는 향후 창의적인 문제 해결 학습이 가능한 에듀테인먼트 콘텐츠 개발의 토대가 될 것이라 생각하기 때문이다. 이후 본 논문에서는 STEAM 융합기반의 코딩교육인 ART 중심의 프로젝트 기반 코딩교육을 ‘코딩아트(cording art) 교육’이라 정의하고자 한다.

1. STEAM 기반의 코딩교육

STEAM(Science, Technology, Engineering, Arts, Mathematics; 과학기술-예술융합의 인재교육) 교육은 과학기술에 대한 학생들의 흥미와 이해도를 높이고, 과학기술 기반의 융합적 사고와 문제 해결 능력을 배양하는 교육이라고 할 수 있다(한국과학창의재단, 2012). 국내에서는 2011년에 최초로 초·중등과정에 도입되었으며, 2013년 11월 한국유아교육학회의 워크숍을 통해 유아대상 STEAM 교육에 대한 관심이 공론화되기 시작하였다. 유아교육은 주제중심의 통합교육을 주요 원리로 하여 국가 수준의 교육과정이 제시되고 있기 때문에 STEAM 교육의 적용이 크게 어렵지는 않다(이연승, 2013).

교육과학기술부(2011)는 STEAM 교육을 통해 ‘지식을 왜 배우는 지, 어디에 사용하게 되는지 이해하고, 실생활에서의 문제 해결력을 배양할 수 있다’고 말한다(김민정, 조형숙, 김대옥, 2014). 김시내(2018)는 기술과 예술의 융합으로 새로운 혁신을 이루어내는 MIT Media Lab의 다양한 프로젝트들이 현재 STEAM 교육의 모델이 된다고 보고 있는데, 그 근거로 John Maeda교수가 말한 “예술과 디자인은 인간공학 기술에 필수적이다”와 “예술은 비판적 사고와 다양한 대안적 사고를 이끌어낸다는 점에서 중요하다”고 설명한다. 또한 소프트웨어 교육이 STEAM 융합교육 실천에 대한 구체적 방법을 제공할 수 있다고 말한다. 예를 들어, T(Technology), A(Art), M(Math)의 결합에서 드로잉 소프트웨어는 간단한 수학적 연산의 적용으로 예술작품을 만드는 환경을 제공할 수 있고, T(Technology), E(Engineering), A(Art)는 피지컬 컴퓨팅을 통해 다양한 방식으로 결합될 수 있는데, 키네틱 오브제를 만들거나 새로운 발명품을 만들어냄으로써 예술과 공학, 기술 사이를 연결할 수 있다(김시내, 2018).

2. ART 중심의 프로젝트 기반 코딩교육

예술심리학자인 Rudolf Arnheim은 시각적 사고(Visual Thinking, 1969)에서 미술활동은 인지활동의 일환으로 감각경험을 포함하며, 지각하기(perceiving)와 생각하기(thinking)가 구분될 수 없을 정도로 얽혀있는 일종의 추리(reasoning)라고 말한다. 시각적 사고는 미술과 심리학은 물론 모든 창의적인 문제해결 분야에 관계하며, 컴퓨팅 사고에서 함양코자 하는 능력과 유사한 지점을 공유한다. 시각적 사고는 컴퓨터 소프트웨어를 통해 그래픽 이미지나 동영상을 만들고, 편집하는 과정에도 필요하며, 추상화 능력, 절차적으로 사고하는 능력, 문제해결 능력과도 연결된다. Mitchel Resnick이 코딩교육용 소프트웨어인 스크래치를 개발할 때 멀티미디어를 활용한 게임이나 애니메이션을 만들 수 있고, 스토리텔링을 담아낼 수 있는 툴로 접근한 것과도 일맥상통한다.

조경미(2017)는 메이커 교육에 기반을 둔 유아과학교육 프로그램의 교수-학습방법에 질문기반 및 프로젝트 기반의 접근법과 만지작거리기(Tinkering), 디자인 사고(design thinking) 등을 적용하였는데, 이를 통해 유아들이 스스로 일상생활의 문제를 해결할 수 있었고, 더 나아가 새로운 경험을 하고 싶도록 독려할 수 있었다. 또한 프로그래밍 교육의 핵심은 문제해결력을 함양시키는데 있다고 주장하는 Wang, Huang, & Hwang(2014)은 문제해결(problem-solving)형 시나리오의 프로젝트 기

반 학습법(project-based learning)을 사용한 스크래치 교육 연구를 진행함으로써 프로젝트 중심 교육이 학생들의 문제해결력을 향상시키는 효과가 있음을 입증하였다.

3. 코딩 아트(coding art) 교육과정의 설계

본 논문에서는 유아대상의 코딩교육에 프로젝트 중심의 교수학습으로 접근하는 STEAM 융합 커리큘럼을 적극적으로 모색하고자 하며, ART 중심의 프로젝트 기반 코딩교육인 ‘코딩아트 교육’의 커리큘럼을 제시하고자 한다. 한국과학창의재단(2012)은 융합인재교육이 제대로 실천되기 위해서는 상황 제시, 창의적 설계, 감성적 체험의 3단계 STEAM 학습준거 틀에 따라야 한다고 제시하고 있다. 따라서 유아 및 초등 저학년을 대상으로 한 코딩아트(coding art) 교육과정을 설계함에 있어서 먼저, 학생에게 실생활과 연계된 문제 해결의 필요성을 구체적으로 느낄 수 있도록 상황을 제시하여 교육에 흥미를 가지고 몰입할 수 있게 해야 한다. 두 번째, 학생이 스스로 문제 해결 방법을 찾아가는 학생 중심의 창의적 설계와 아이디어 실현, 과정을 중요시하는 다양한 산출물 도출과 협력학습을 통해 문제를 해결할 수 있게 해야 한다. 마지막으로, 학생이 스스로 문제를 해결한 성공의 경험을 직접 체험함으로써 다시 새로운 도전이 가능할 수 있도록 도와야 한다. 이렇게 창의적으로 문제를 해결하는 프로젝트 중심의 코딩 아트 교육이 제대로 이루어지기 위해서는 앞서 연구한 디자인 사고의 프로세스가 적용되도록 교육과정이 설계되어야 할 것이다.

따라서 본 논문에서 제안하는 유아용 코딩 아트 교육의 교육과정은 3단계 STEAM 학습준거 틀에 따라 첫 번째 상황제시 단계에서는 학습할 교육내용에 대한 짧은 동영상 시청하고, 두 번째 창의적 설계 단계에서는 각각의 학습내용과 연계된 언플러그드 형태의 보드게임을 통해 재미있게 놀면서 핵심적인 개념을 이해한 후, 세 번째 감성적 체험 단계에서는 미술 프로젝트 활동을 통해 배운 내용을 응용하여 자신만의 창의적인 결과물들을 만들어 보는 학습을 진행하고자 한다.

즉, 코딩아트 교육의 교육과정은 크게 컴퓨터과학과 프로그래밍, 컴퓨팅 사고력, 디자인 사고력의 4가지 영역을 모두 포함하도록 교육내용을 구성하였는데, 학습자의 발달단계에 따라 학습의 깊이와 양, 난이도를 조절할 수 있도록 학습내용의 확장성을 염두에 두고 범주를 설정하였다. 유아 및 초등 저학년 학생들이 학습해야 하는 학습내용을 구성하기 위해서 <표 2>에서 정리했던 컴퓨팅 사고 및 디자인 사고력의 구성요소를 바탕으로 교육내용의 키워드를 추출하였고, 이를 통해 학습영역을 크게 컴퓨터 과학(데이터)의 개념과 컴퓨팅 사고의 구성요소, 디자인 사고의 구성요소, 알고리즘과 프로그래밍 언어의 이해로 구분하여 보았다. 또한 각각의 영역에 따른 세부 학습내용은 추출한 키워드를 중심으로 총 25개로 선정하였는데, 컴퓨터 과학(데이터) 영역에서는 이진수, 컴퓨터 용어, 텍스트 압축, 명령어, 이미지 자료(색, 형태, 도형) 등의 5개 학습내용을 선정하였고, 컴퓨팅 사고 영역에서 순차와 절차, 규칙과 패턴, 분해와 반복, 조건과 연산으로 8개의 학습내용을 선정했으며, 프로그래밍 영역에서는 알고리즘과 프로그래밍 언어로 2개의 학습내용을 선정하였고, 디자인 사고 영역에서는 기억, 분류, 유추, 추론, 조합, 분석, 연상, 표현, 전략, 협력 등의 10개 학습내용으로 총 25개의 학습내용을 구성하였다.

표 7. 유아용 코딩아트 교육의 교육과정

영역	학습 내용	개념이해 보드게임 활동	미술 프로젝트
컴퓨터과학 (데이터)	이진수 컴퓨터 용어	팝콘(Pop con) 요원12	픽셀 드로잉 캐릭터 카드

	텍스트 압축 명령어 이미지	라온, ZIP Go and Stop 디지오스, 우봉고, 테트리스링크	암호문 풀기 미로와 길찾기 동물농장
컴퓨팅 사고	순차/절차 규칙/패턴 분해/반복 조건/연산	러시아워, 스택버저, 플레이로봇 우노, 큐비츠(Qbits) 맛있는 코딩, 시그널 코딩버스, 루미큐브, 스도쿠	생일파티 준비 목걸이와 팔지 픽셀레이션 벽지 미래의 직업
프로그래밍	알고리즘 프로그래밍 언어	엔트리봇, 코드마스터 Catch the Cat, Catch the Dog	지도 그리기 정보이야기
디자인 사고	기억/분류 유추/추론 조합/분석 연상/표현 전략/협동	Rat a Tat Cat, 파라오, 세트 초콜릿픽스, 클루, 다빈치코드 Figurix, 로봇얼굴맞추기 텔레스트레이션, 딕싯, 콘셉트 Take it Easy, 애니그마 와글와글턴전	냉장고 정리 탐정수첩 투명 컵 귀신 상상 놀이터 맛있는 주스광고

먼저, 1단계 상황제시 단계에서는 ‘짧은 동영상 보기 과정’으로 아이들이 좋아할 만한 캐릭터가 등장하는 5분짜리 애니메이션으로 학습할 교육내용에 대한 몰입을 유도하는 도입용 스토리텔링을 설계하였다. 그리고 2단계 창의적 설계 단계에 해당하는 논리적인 사고력을 증진시키기 위한 개념을 이해하는 과정에서는 각각의 영역별 학습내용에 맞추어 개념을 쉽고 재미있게 이해할 수 있도록 연플러그 형태의 ‘보드게임을 접목한 놀이 활동’으로 교육과정을 설계하였는데, 개별 또는 2-4명으로 구성된 팀 활동을 통해 경쟁과 협력, 사회성을 동시에 배울 수 있도록 하였다. 끝으로 3단계 감성적 체험 단계에서는 창의적인 문제해결 능력을 증진시키기 위한 과정으로 배운 내용을 응용하여 ‘미술 프로젝트로 표현해보는 창작 활동’을 교육과정으로 설계하였다. 이를 바탕으로 6-7세를 대상으로 하는 코딩아트 교육과정의 실제 사례를 <표 8>로 제시하였다.

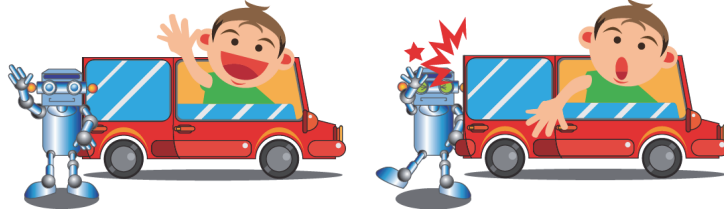
표 8. 6-7세 대상 코딩아트 교육과정의 사례

학습주제	절차적 사고
용어이해	컴퓨터와 대화하기 위해서는 언어를 컴퓨터에 입력해 줘야 한다. 그런데, 컴퓨터는 아무 언어나 이해하지 못하고, 단계별 순서에 따라 논리적인 언어만 이해할 수 있다. 컴퓨터가 “절차적 사고”를 하기 때문이다.
학습개요	컴퓨터와 대화하는 방법인 절차적 사고에 대해 이해하고, 순차와 논리에 맞게 명령어를 사용해 본다.
학습목표	1. 순차와 규칙을 이해할 수 있다. 2. 논리적 사고력을 키울 수 있다. 3. 절차적 사고에 따른 코딩(프로그래밍)을 이해할 수 있다.
준비물	자료카드, 보드게임 자료, 미술도구
도입 (개념정리)	▶ 오늘은 컴퓨터와 대화하는 방법인 절차적 사고에 대해 공부합니다. - 컴퓨터는 똑똑해서 동시에 여러 가지 일을 해결하는 것 같지만, 순서대로 하나씩 일을 처리해 나가요. 다만, 속도가 굉장히 빠르죠. 컴퓨터는 스스로 판단을 바꾸지 못하고 지시대로만 행동하기 때문에, 순서와 논리에 맞게 명령을 해야 해요.
5분	예를 들어, “문 닫고 방 안으로 들어와” 라는 말을 듣고, 순서에 따라 행동한다면, 문을 닫은 후에는 방 안으로 들어오지 못하게 되므로, 이 말은 논리적이지 못하며, “방 안으로 들어와”라는 명령은 따를 수 없게 됩니다. 따라서 컴퓨터에게 명령을 전달하기 위해서는, 방 안으로 들어 온 후에 문을 닫도록, 정확하게 지시해야 합니다. 다시 말해서, 언어를 컴퓨터에 입력할 때는, 절차적 사고에 따라 입력해야 하며, 이것을 코딩 또는 프로그래밍이라고 합니다.

▶ 절차적 사고 예시(그림카드)

“어서 자동차에 타”라고 말을 했을 때, “문을 연 후에 자동차에 타”라는 지시 절차를 빼먹었다면, 컴퓨터는 자동차에 타지 못하고 닫힌 문에 부딪치게 돼요.

“자동차에 타”라는 명령어에만 충실했기 때문이죠. 그렇다면, 논리에 맞게 순서를 제대로 바로잡아 봅시다. 어떻게 명령해야 할까요? (발표하기)



▶ 절차적 사고를 배울 수 있는 보드게임을 해 봅시다.

개념설명
(이론이해)

● 게임 내용

절차적 사고를 하는 로봇이 피자배달을 하러 출발을 하려고 해요. 피자 가게를 출발하여, 문 앞에 작은 나무 한그루가 있는 건물로 배달을 할 거예요. (보드게임 판 참조) 그런데 우리가 로봇을 조종해야 해요.

35분

● 게임 규칙

- 1) 로봇은 산, 바위 등의 장애물은 넘지 못하며, 도로만 이용할 수 있어요.
- 2) 명령어 카드를 이용해야만 피자배달 로봇을 조종할 수 있어요.
- 3) 조종실에서 명령어 카드를 한 단계씩 순차적으로 입력하여 로봇을 조종해 봅시다.

● 게임 준비물

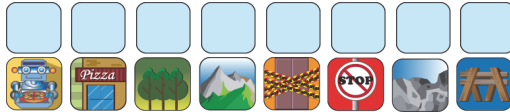
명령어 카드	배달로봇(말)	보드게임 판
- 앞으로 한 칸 가기 - 왼쪽으로 90° 방향 바꾸기 - 오른쪽으로 90° 방향 바꾸기 		

▶ 절차적 사고를 배울 수 있는 보드게임을 만들어 봅시다.

- 앞서 진행했던 게임과 동일한 내용과 규칙의 게임을 스스로 만들어 봅시다.

● 게임 보드의 칸 안의 내용을 미술활동을 통해 직접 꾸며 봅시다.

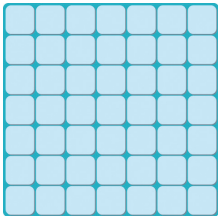

배달로봇(말), 출발지, 도착지, 장애물(산, 바위, 도로표지판) 등의 그림을, 빈 카드마다 각각 그려 넣습니다. (예시 자료를 보고 참고하기)







미술활동
(실습이해)

● 그림을 그려넣은 카드를, 보드의 빈 칸에 채워 넣어 봅시다.

35분

빈 보드	보드 꾸미기 완성 예시
	

- 조종실 빈 칸에 명령어 카드를 입력하여 해결과제를 완성합니다.

	명령어 카드	조종실
입력전		
입력후		

- 생각 확장

가장 빨리 배달할 수 있는 길을 찾아봅시다.

명령어를 적게 써서 도착한다면, 더 빨리 도착할 수 있겠죠?

▶ 오늘 실습에 대한 간략한 리뷰 및 발표시간을 갖는다.

정리

1. 절차적사고의 개념을 다시 정리한다.

- 컴퓨터는 지시한 것만 행동해요.

5분

- 컴퓨터와 대화할 때는 순차와 논리에 맞게 이야기해야 해요.

2. 각 학습자에게 발표의 기회를 주어 이해도를 평가한다.

3. 실습장소를 정리 한다.

V. 결론 및 제언

본 연구는 유아(5-7세) 및 초등 저학년(8-9세)을 대상으로 하는 코딩교육의 교육과정을 STEAM 융합교육이자, ART 프로젝트 기반의 창의적인 문제해결 방식인 코딩아트 교육으로 제안하였다. 유아 및 초등 저학년을 대상으로 하는 코딩교육은 컴퓨터를 사용하는 프로그래밍 언어 교육보다는 언플러그드 형태의 STEAM 융합교육이 더욱 효과적이며, 소프트웨어 활용 능력보다는 컴퓨팅 사고력(Computational Thinking)과 디자인 사고력(Design Thinking)을 키우기 위한 교육목표를 가지는 것이 바람직하기 때문이다. 이러한 교육목표 아래 컴퓨터 과학의 개념과 컴퓨팅 사고력, 디자인 사고력을 키우기 위한 총 25개 항목의 학습내용을 구성하였으며, 보드게임을 활용한 개념이해 활동과 함께 이해한 학습내용을 응용하여 프로젝트 기반의 창의적인 작품으로 만드는 미술활동으로 교육과정을 구성하여 보았다. 또한 제안한 코딩아트 교육과정을 토대로 6-7세를 대상으로 하는 코딩아트 교육과정의 실제 사례를 제시하였다.

이를 바탕으로 향후 후속연구에서는 유아(5-7세) 및 초등 저학년(8-9세)을 대상으로 한 코딩교육의 교육과정을 실제로 운영할 수 있는 에듀테인먼트 콘텐츠를 개발하고자 하며, 더 나아가 개발된 교육 프로그램을 실제 교육 현장에 적용해 봄으로써 그 교육적 효과를 검증해보고자 한다.

참고문헌

- 경남신문(2017, 8월 10일). 놀면서 배우는 '코딩' 창의력 쑥쑥.
- 구영은(2015). 초등학교 저학년 학생을 대상으로 한 놀이학습 기반 언플러그드 교육의 효과성 분석. 미간행 석사학위논문, 경인교육대학교 교육전문대학원.
- 김민정, 조형숙, 김대옥(2014). 국내 초등학교 STEAM 교육 연구 현황분석을 통한 유아교육에서의 방향 탐색. *유아교육연구*, 34(4), 139-161.
- 김성환(2017). SW코딩교육이 초등과학영재와 일반학생의 진로성숙도, 사회적 기여의식에 미치는 영향. 미간행 석사학위논문, 경인교육대학교 교육전문대학원.
- 김시내(2018). 소프트웨어 사회, 빅데이터 그리고 미술교육. *예술교육연구*, 16(3), 37-50.
- 김자인(2015). 디스쿨(d.School)의 디자인사고 교육. *디지털디자인학연구*, 15(4), 97-108.
- 김진옥(2018). 메이커 기반 STEAM 교육을 위한 수업 모형 개발. 미간행 박사학위논문, 한국교원대학교 대학원
- 김형식(2015). 보드게임기반 수학 STEAM 교육 프로그램이 융합인재역량에 미치는 영향. 미간행 석사학위논문, 경인교육대학교 교육전문대학원.
- 박성립(2016). 소프트웨어 설계 능력 향상과 컴퓨팅 사고력의 관계. 미간행 박사학위논문, 성균관대학교 대학원
- 사이언스타임즈(2019, 1월16일). 4차 산업혁명 시대, 코딩 교육 방향은?
- 송상수(2015). 소프트웨어 교육. https://blog.naver.com/gi_sik_in/220230820622
- 오경선, 안성진(2016). 소프트웨어 교육을 위한 컴퓨팅사고 교육내용 설계 기본 연구. *한국컴퓨터교육학회 논문지*, 19(2), 11-20.
- 윤혜진(2018). 디자인사고 기반 메이커교육 모형 개발. 미간행 박사학위논문, 경희대학교 대학원.
- 이경진, 이지현(2017). 유아의 실행기능이 수학적 문제해결력에 미치는 영향. *어린이문학교육연구*, 18(1), 311-334.
- 이도현(2015). 집단 창의성 교육을 위한 방안으로서 과학 교육에 디자인적 사고를 도입하기 위한 핵심 역량 탐색 및 프로세스 개발 연구. 미간행 박사학위논문, 한국교원대학교.
- 이승철(2018). 디자인적 사고 기반 메이커 교육 프로그램이 초등학생의 창의적 문제해결력 및 학습동기에 미치는 영향. 미간행 석사학위논문. 한국교원대학교 대학원.
- 이애화(2018). 국내 소프트웨어교육 연구동향 분석. *교육정보미디어연구*, 24(2), 277-301.
- 이연승, 성현주(2017). 코딩용 로봇, 비봇(Bee-Bot)을 활용한 수학적 문제해결력 증진 프로그램 개발 및 효과. *어린이미디어연구*, 16(3), 261-281.
- 이은경(2009). Computational Thinking 능력 향상을 위한 로봇프로그래밍 교수학습 모형. 미간행 박사학위논문, 한국교원대학교 대학원.
- 이현민(2018). 듀이 미학 관점에서 창의적 소프트웨어 교육에 대한 소고: 아트코딩. *문화예술교육연구*, 13(2), 1-20.
- 이현진(2018). 그래픽과 멀티미디어 사고를 접목한 어린이 코딩교육 융합 커리큘럼 개발 연구. *디자인융합연구*, 17(1), 63-82.
- 정인기(2016). Computational Thinking에서의 추상화 개념에 대한 고찰. *정보교육학회논문지*, 20(6), 585-596.
- 전용주(2017). 새로운 교육과정의 소프트웨어 교육을 위한 컴퓨팅 사고력 기반 창의적 문제해결

- (CT-CPS) 수업모형의 개발 및 적용. 미간행 박사학위논문, 한국교원대학교 대학원.
- 조경미(2017). 메이커 교육(Maker Education)에 기반을 둔 유아과학교육 프로그램 개발 및 효과. 미간행 박사학위논문, 경성대학교 대학원.
- 최정아(2016). 중등 예비미술교사의 디자인 교육을 위한 디자인 사고 프로세스 적용 가능성 연구. **학습자 중심교과교육연구**, 16, 1357-1379.
- 하희정(2017). SW코딩기반 메이커교육용 교수학습모형 개발. 미간행 석사학위논문, 경인대학교 교육전문대학원.
- 한국과학창의재단(2012). **손에 잡히는 STEAM교육**. 서울: 한국과학창의재단.
- 한국교육개발원, 한국교육학술정보원(2015). SW교육 교수학습 모형 개발 연구. 2015년 **교육정책네트워크 교육현장지원연구**, 57~78.
- 한선관(2017). 컴퓨팅 사고 신장을 위한 놀이중심 SW교육 교수학습 전략. **정보교육학회논문지**, 21(6), 657-664.
- 한선관, 신수범(2011). 언플러그드 에듀테인먼트 교육프로그램의 개발. **한국정보교육학회 논문지**, 15(2), 201-208.
- Arnheim, R. (2004). **시각적 사고(Visual Thinking)**. 김정오 역(2004), 서울:이화여자대학교 출판부.
- Brown, T.(2008). Design Thinking. *Harvard Business Review*, June, 84-94.
- Carroll, M., Goldman, S., Britos, L., Koh, J., Royalty, A., & Hornstein, M. (2010). Destination, imagination, and the fires within: Design thinking in a middle school classroom. *International Journal of Art & Design Education*, 29(1), 37-53.
- IDEO(2011). Design Thinking Toolkit for Educators. (<https://designthinkingforeducators.com/>)
- Martin, R. (2009). **디자인 씽킹(Design thinking)**. 이견식 역(2009), 서울:웅진 윈즈.

ABSTRACT

The Curriculum Design for kid's Coding Art Education

Ahn, Seong-Hye(Sangmyung University)

Modern society is changing into a convergence technology society based on software. The importance of computational thinking (CT) has been emphasized in order to nurture the creative convergence talents needed in the fourth industrial revolution era. The purpose of this study is to design a curriculum for coding education to cultivate computational thinking for preschool children (5-7 years old) and elementary school students (8-9 years old) who are exposed to coding education for the first time. First, I tried to set the direction of coding education that reflects characteristics of preschool children and elementary school students. To do this, I analyzed the teaching - learning model of the coding education and examined the educational effects by the case study of the kid's coding education contents. As a result, there are many studies that the unplugged STEAM convergence education and creative problem solving focused project education are effective for kid's coding education. It is recommended that children's coding education should have educational goals to develop computational thinking (CT) and design thinking (DT). Therefore, I would like to propose a project-based unplugged coding art education model, and a STEAM education that integrates mathematics, science, technology, and art activities. Based on the educational goals, I constructed 25 learning contents for understanding of computer science and for developing computer thinking and design thinking. In the end, I designed a project-based coding art curriculum that expresses as a creative artwork at the same time as a conceptual understanding activity using a board game so that students can learn fun and active.

Key words : coding education, coding art, computational thinking, design thinking, STEAM education, project-based learning

접수일 : 2019년 4월 30일

심사완료일 : 2019년 6월 3일

게재확정일 : 2019년 6월 17일