교육학석사 학위논문

수학과 코딩을 융합하는 자유학기제 교육과정 디자인 및 적용 연구

2022년 8월

서울대학교 대학원

수학교육과

강 하 람

수학과 코딩을 융합하는 자유학기제 교육과정 디자인 및 적용 연구

지도교수 유 연 주

이 논문을 교육학석사 학위논문으로 제출함
2022년 08월

서울대학교 대학원
수학교육과
강 하 람

강하람의 석사 학위논문을 인준함
2022년 08월

위 원 장 이 경 화 (인)

부위원장 조 한 혁 (인)

위 원 유 연 주 (인)

국문 초록

본 연구는 자유학기제의 목적에 맞게 학생들이 꿈과 끼를 찾으며 미래 역량을 함양할 수 있도록 수학과 정보의 코딩을 융합하는 자유학기제 교육과정을 디자인하고 적용한 결과를 살펴보는 것에 연구의 핵심이 있다. 본 연구는 교육부에서 4차 산업혁명 시대에 필요한 인간상으로 2015 개정 교육과정에서는 창의·융합형 인재, 2020 개정 교육과정에서는 창의성을 갖춘 주도적인 사람을 강조함에 따라 서로 다른 지식을 융합할 필요성에 의해 연구를 실시하였다. 중학교 정보 교과의 필수, 고등학교 인공지능 수학 과목의 도입에 맞춰 문자와 치환, 알고리즘과 순서도, 변수와 함수, 기하, 공간도형을 학습하는 것을 지향하여 3차원 좌표계 기반 코딩 환경으로 수학과 코딩을 융합할 수 있는 터틀크래프트 학습 환경을 제시한다. 먼저, 터틀크래프트를 이용하여 수학과 코딩을 융합한 자유학기제 교육과정의 디자인 원리를 논하였다. 다음으로 디자인한 교육과정의 효과적인 적용을 위해 최소 코딩게임 교수전략을 활용하고 그에 따른 학생들의 학습 변화과정을 분석하였다. 마지막으로, 자유학기제 수업에서 수학과 코딩을 융합한 교육과정을 통해 학생들의 진로 탐색에 대한 태도와 수학 및 코딩에 대한 인식 변화를 살펴보았다.

이를 위해 본 연구에서는 서울시 소재의 S 중학교 1학년 학생 35명을 대상으로 ‘코딩수학’ 수업을 진행하였다. ‘코딩수학’은 중학생을 대상으로 하는 자유학기제 교육과정에서 실시하는 수업으로 코딩과 수학을 융합한 내용을 다룬다. ‘코딩수학’ 수업에서 최소 코딩게임 교수전략을 적용했을 때, 학생들의 학습 변화과정을 분석하기 위해 적용하기 전과 적용한 후의 학생들의 명령어를 활동지를 이용하여 수집하였다. 수집한 학생들의 명령어 변화과정을 유형화하여 비교하고 명령어를 사용할 때 학생들의 핵심 사고 과정을 분석하였다. 또한, ‘코딩수학’ 수업을 통해 학생들의 진로 탐색에 대한 태도와 수학 및 코딩에 대한 인식 변화를 살펴보기 위해 사전 설문조사, 사후 설문조사, 사후 인터뷰를 실시하였다.

분석 결과, 수학과 코딩을 융합하는 코딩수학 교육과정에서 최소 코딩 게임 교수전략을 적용하였을 때, 수준이 높은 학생과 수준이 낮은 학생 모두 학습 동기를 유발하고 수준 상승을 유도하는 데 효과적임을 확인하였다. 또한, 수학과 코딩을 융합하는 자유학기제 교육과정을 통해 학생들의 진로 탐색과 수학 및 코딩의 인식에 바람직한 변화가 있었음을 확인하였다. 본 연구는 수학과 코딩을 융합하는 자유학기제 교육과정 디자인 및 적용하는 연구를 통해 자유학기제의 목적을 달성하면서 창의·융합형 인재를 양성할 수 있는 교육과정의 방향을 제시할 것으로 기대한다.

주요어 : 자유학기제, 융합 교육과정, 최소 코딩게임, 알고리즘, 수준 상승, 학습 동기

학 번 : 2018-23405

목 차

표 목 차

그 림 목 차

Ⅰ. 서 론

1. 연구의 목적 및 필요성

우리나라는 2015 개정 교육과정을 도입하면서 학생들이 미래 핵심역량을 함양할 수 있도록 노력하고 있으나, 수학에 대한 흥미 및 행복 지수가 낮은 상황이다. TIMSS 2011에서는 중학교 2학년 수학 성취도 결과가 42개국 중 1위임에도 불구하고 수학에 대한 흥미를 묻는 문항에서 '좋아함'의 응답자 비율이 8%밖에 되지 않았으며, '좋아하지 않음'의 응답자 비율이 무려 56%나 되었다. 수학에 대한 가치 인식을 묻는 문항에서는 '가치 있음'의 응답자 비율이 14%밖에 되지 않았으며, '가치 없음'의 응답자 비율이 34%나 되었다. 또한, 청소년 행복 지수를 묻는 문항에 대한 점수가 OECD 23개국 중 23위인 최하위에 해당하였으며, 청소년 행복 지수 국제 비교 연구에서 OECD 국가 중 교육은 상위 수준이지만 주관적 행복이 하위 수준에 속하는 그룹에 유일하게 속하는 것으로 나타났다(교육부, 2015b; 박종일 외, 2010).

이를 위해 우리나라에서는 학생들의 꿈과 끼를 키우는 행복 교육으로 변화시키는 전기를 마련하기 위해 자유학기제를 2016년부터 모든 중학교에서 시행하고 있다(교육부, 2015c). 또한, 2018년부터는 희망하는 학교에서 자유학기제를 자유학기제로 확대하여 운영하였으며 2020년부터는 모든 중학교에서 자유학기제를 시행하고 있다(교육부, 2017). 자유학기제의 목표는 자신의 적성과 미래에 대해 탐색하고 꿈과 끼를 찾으며 창의성, 인성, 자기주도적 학습 능력 등 미래 핵심역량을 함양하여 행복 교육을 실현하는 것이다. 이에 2022 개정 교육과정부터는 자유학기에서 예술 체육, 동아리 활동을 제외한 주제 선택 및 진로 탐색 활동만을 운영하며, 진로연계 학기를 도입하여 진로 탐색 활동을 강조하고 있다.

그러나 자유학기제의 목표를 달성하는데 학생들은 자유학기가 끝난 후 일반학기로의 연계 부족(45.3%), 자유학기에 대한 자세와 인식 부족(24.7%), 진로 탐색과 연관된 진로 체험 프로그램 부족(21.3%)을 문제점으로 삼았다(백은미, 2018). 또한, 자유학기제에 받고 싶은 수업의 유형에 대한 학생 수요조사에서 학생들은 체험 중심수업을 선호하는 학생이 65.49%로 가장 많았다(한국교육개발원, 2013). 따라서 자유학기제의 목표에 맞게 학생들의 꿈과 끼를 찾을 수 있도록 진로 탐색과 연관된 교육과정, 학습 동기유발 및 창의력과 자기주도적 학습 능력을 기를 수 있는 체험 중심 교육과정, 자유학기가 추후에 일반학기와 연계될 수 있는 교육과정을 디자인할 필요가 있다.

한편, 4차 산업 시대가 도래함에 따라 전 세계적으로 SW 교육(소프트웨어교육)의 중요성이 대두되고 있다. 이에 세계의 여러 나라는 SW 교육을 교육과정으로 도입하며 단순 코딩 능력을 넘어 컴퓨팅 사고력 교육에 힘쓰고 있다(권오남, 2018). 2015 개정 교육과정에서는 학생들이 소프트웨어에 대한 기초소양을 충실히 갖추어 나갈 수 있도록 중학교에서 SW 교육 중심의 정보 교과를 필수 과목으로 지정하였다(교육부, 2015a). 또한, 교육부(2020)는 4차 산업 시대를 살아갈 학생들이 정보지능기술을 활용할 수 있도록 인공지능을 학교 교육에 적극 도입하기로 하며 2021년 2학기부터 고등학교에 '인공지능 수학' 과목을 선택할 수 있도록 하였다. 2022 개정 교육과정에서는 초·중·고 전반에 걸쳐 SW 교육을 내실화하기 위해 모든 교과에서 디지털 기초소양을 강화하고 있으며, 중학교에서 교과 시간을 활용한 정보 교육을 확대하기로 하였다(교육부, 2022a).

교육부(2015a)에서는 SW 교육의 중요성과 더불어 2015 개정 교육과정에서 추구하는 미래의 인간상으로 '창의·융합형 인재'를 강조하였으며, 교육부 장관은 2015 개정 교육과정의 개정 배경으로 '유연하고 창의적인 사고력, 서로 다른 지식을 융합하고 활용할 수 있는 창의·융합형 인재 양성'을 말하였다. 또한, 2022 개정 교육과정에서는 추구하는 인간상으로 '창의성을 갖춘 주도적인 사람'을 강조하였으며, 여기서 창의와 혁신은

문제해결, 융합적 사고, 도전 의식을 갖춘 사람을 말한다(교육부, 2022a). 이처럼 창의·융합형 인재 및 창의성을 갖춘 주도적인 인재를 양성하기 위해서는 서로 다른 지식을 융합하는 교육과정을 개발할 필요성이 있다.

이에 본 연구에서는 자유학기제의 목적에 맞게 자신의 적성과 미래를 탐색하고 4차 산업혁명 시대에 필요한 미래 핵심역량을 함양할 수 있도록 수학과 코딩을 융합하는 자유학기제 교육과정을 디자인하고 그 원리를 분석하고자 한다. 디자인하는 원리는 다음과 같다. 첫째, 창의력 사고력, 수학적 사고력, 컴퓨팅 사고력을 함양할 수 있도록 학생들의 수준 상승이 일어나도록 하는 교육과정이어야 한다. 둘째, 정보 교과, 인공지능 수학 과목 도입에 맞춰 수학 교과와 정보 교과의 내용 체계에 있는 공통 요소를 바탕으로 문자와 치환, 알고리즘과 순서도, 변수와 함수 등을 융합하는 교육과정이어야 한다. 이때 기하, 공간도형을 학습하는 것을 지향하여 3차원 좌표계 기반 코딩환경에서 코딩과 수학을 융합할 수 있는 '터틀크래프트'라는 코딩환경을 사용한다.

다음으로 본 연구에서는 디자인한 교육과정의 효과적인 적용을 위해 최소 코딩게임 교수전략을 활용하고 그 결과를 분석하고자 한다. 자유학기제에서는 평가 결과가 점수로 표기되는 지필평가 위주의 중간고사나 기말고사를 실시하지 않는다. 학생의 성장을 끌어내기 위해서는 수업 활동과 연계된 다양한 유형의 평가를 실시하여 학생의 부족한 부분을 확인하고 그에 맞는 학습 활동을 제공해야 한다(경상남도교육청, 2021). 평가의 부재는 학생들의 학습 동기가 효과적으로 유발되지 못할 수 있다. 이에 학생들의 학습 동기를 효과적으로 유발하기 위한 최소 코딩게임 교수전략을 자유학기제 수업에 적용하고 학생들의 학습 변화과정을 분석하여 그 효과성을 확인하고자 한다.

마지막으로 디자인한 자유학기제 교육과정이 본래 자유학기제의 목적과 취지인 자신의 적성과 미래를 탐색을 일어나도록 하는지 분석하고자 한다. 이를 위해 디자인한 교육과정을 자유학기제 수업에 적용하고 학생들의 진로 탐색에 대한 태도, 수학과 코딩에 대한 인식 변화를 살펴본다.

본 연구에서 디자인한 교육과정과 최소 코딩게임 교수전략은 강하람 외(2021)가 2020학년도 1학기 및 2021학년도 1학기에 시흥시 소재의 초등학교 6학년 학생들을 대상으로 적용하였다. 그 결과를 바탕으로 초등학생을 대상으로 하는 수학과 코딩을 융합한 코딩수학 교육과정이 개발되었다. 본 연구에서는 초등학생을 대상으로 하는 교육과정을 중학생을 대상으로 자유학기제 교육과정에서 적용할 수 있도록 교육과정 내용을 수정·보완하여 수학과 코딩을 융합하는 자유학기제 교육과정을 개발하였다. 마찬가지로 초등학생을 대상으로 하는 최소 코딩게임 교수전략도 중학생 수준에 맞게 추가된 코딩 명령어를 고려하여 중학교 자유학기제 교육과정에서 적용할 수 있도록 차시별로 구체적인 단계와 방법을 제안하도록 수정·보완하였다.

2. 연구 문제

이상 언급한 바와 같이, 본 연구에서는 3차원 좌표계 기반 코딩환경인 터틀크래프트를 활용하여 수학과 코딩을 융합하는 자유학기제 교육과정과 최소 코딩게임 교수전략을 디자인하였다. 이를 2022학년도 1학기에 서울시 소재의 중학교 1학년 학생 40명을 대상으로 자유학기제 수업에 실제로 적용한다. 먼저, 수학과 코딩을 융합하여 학생들의 수준 상승이 일어나도록 하는 융합 교육과정의 원리를 분석한다. 다음으로, 최소 코딩게임 교수전략을 적용하기 전후, 학생들의 학습 변화과정을 통해 최소 코딩게임 교수전략의 효과를 살펴본다. 마지막으로, 수학과 코딩을 융합하는 자유학기제 교육과정을 통해 학생들의 진로 탐색에 대한 태도, 수학과 코딩에 대한 인식에 어떤 변화가 있었는지 살펴본다. 이러한 목표를 달성하기 위해 연구 문제를 다음과 같이 설정하였다.

첫째, 3차원 좌표계 기반 코딩환경을 활용하여 수학과 코딩을 융합하는 자유학기제 교육과정을 어떤 원리로 디자인할 수 있는가?

둘째, 디자인된 교육과정을 현장에 적용할 때, 학생들의 학습 동기유발을 위해 개발된 최소 코딩게임 교수전략에 따른 학생들의 학습 변화과정은 어떠한가?

셋째, 수학과 코딩을 융합하는 자유학기제 교육과정 기반의 수업을 통해 학생들의 진로 탐색에 대한 태도와 수학 및 코딩에 대한 인식에 어떤 변화가 있었는가?

3. 연구 내용의 개요

Ⅱ장 '이론적 배경'에서는 본 연구를 적용하고자 하는 자유학기제 교육과정의 목적 및 운영을 살펴보고, 2015 개정 교육과정 및 2022 개정 교육과정에서 추구하는 인간상, SW 교육의 변화를 살펴본다. 중학교 정보과의 내용 체계와 중학교 수학과의 내용 체계 및 고등학교 교육과정의 내용 체계에서 알고리즘과 순서도, 변수 등 코딩에서 다양한 수학적 개념이 사용되고 있음이 중·고등학교 전반에 걸쳐 나타나고 있는데, 이러한 수학과 및 정보과 교육과정의 공통 요소를 분석한다. 수학과 코딩의 밀접한 관련성을 바탕으로 3차원 좌표계 기반 코딩환경인 터틀크래프트의 특성에 대해 논의한다. 그 후 터틀크래프트에서 컴퓨팅 사고력의 추상화 과정을 신장시킬 수 있는 최소 코딩게임 교수전략에 대해 살펴보고 최소 코딩게임 교수전략을 통해 효과적으로 유발할 수 있는 학습 동기의 개념 및 요소에 대해 살펴본다.

Ⅲ장 '자유학기제 교육과정 디자인 및 최소 코딩게임 교수전략의 구성'에서는 수학과 정보의 코딩을 융합하는 원리를 분석하고 이를 바탕으로 디자인한 자유학기제 교육과정의 주요 수업 내용을 제안한다. 또한, 자유학기제 교육과정을 효과적으로 적용하기 위해 최소 코딩게임 교수전략을

도입한다. 최소 코딩게임을 학생들의 학습 동기를 유발하고 수준 상승을 일으키도록 구성하고 그 구성원리를 논의한다.

Ⅳ장 '자유학기제 교육과정 및 최소 코딩게임 교수전략의 적용'에서는 자유학기제에서 교육과정 및 최소 코딩게임 교수전략을 적용하고 그 결과를 살펴본다. 설계한 최소 코딩게임 교수전략 적용 문제와 수업 상황을 바탕으로 최소 코딩게임 교수전략을 적용하기 전과 후의 명령어를 비교하여 학생들의 학습 변화과정을 제시한다. 이를 통해 최소 코딩게임 교수전략이 자유학기제 교육과정에서 학생들의 동기를 유발하고 수준 상승을 일으키는 데 효과적으로 적용되고 있는 특징을 논의한다. 또한, 자유학기제의 목적인 진로 탐색에 대한 태도 변화, 수학과 코딩에 대한 인식 변화를 살펴보고 이를 바탕으로 수학과 코딩을 융합하는 교육과정의 교육적 의미에 대해 살펴보고자 한다.

마지막으로 Ⅳ장에서는 자유학기제에서 수학과 코딩을 융합하는 교육과정과 최소 코딩게임 교수전략을 적용했을 때 학생들의 변화를 종합, 요약하고, 본 연구의 의의, 제한점 및 후속 연구를 위한 제언을 담고자 한다.

Ⅱ. 이론적 배경

1. 자유학기제

1.1. 자유학기제의 개념

자유학기제는 자기주도적 학습 능력을 기르기 위해 중학교 1학년 동안 학생 참여형 수업을 실시하고 학생의 소질과 적성을 키울 수 있는 다양한 체험 활동을 중심으로 교육과정을 운영하는 제도이다(교육부, 2017). 다시 말해, 중학교 1학년 동안은 학생들이 시험 부담에서 벗어나 다양한 체험 중심의 학습 활동을 하면서 스스로 학습하고 좋아하는 것을 찾아가는 과정을 통해 미래 사회를 살아가는 데 필요한 역량을 갖출 수 있도록 교육과정을 유연하게 운영하는 제도이다(경상남도교육청, 2021). 자유학기제 교육과정을 통해 학생들은 자신을 이해하며 꿈과 끼를 찾고 스스로 역량을 키울 수 있을 것이다.

교육부(2015c)에서 제시하는 자유학기제의 추진 방향은 다음과 같다. 첫째, 꿈과 끼를 키우는 교육 활동이 원활히 이뤄지도록 학교 교육과정의 자율성을 확대하고 학생 중심 교육과정을 운영한다. 둘째, 자유학기를 중심으로 초등학교에서는 진로를 인식하고, 중학교에서는 진로를 탐색하며, 고등학교에서는 진로 준비 및 설계로 이뤄지도록 진로 교육을 연계·활성화한다. 셋째, 학생이 참여하는 다양한 자유학기 활동을 활성화한다. 넷째, 학생의 성장과 발달을 지원하는 과정 중심의 평가를 실시한다. 다섯째, 자유학기를 교육과정 운영, 수업 방법 개선 등 학교 교육 전반의 변화를 견인하는 계기로 활용한다. 따라서 자유학기제에서는 교육과정이 학생들이 진로를 탐색하고 활동에 참여하는 학생 중심 교육과정으로 운영되어야 한다.

1.2. 자유학기제의 운영

가. 자유학기제 교육과정

2018년부터 약 1,500개교(전체 중학교의 약 46%)에서 자유학기제를 1학년으로 확대하는 자유학기제 운영을 시작으로 점차 확대돼 2020년 모든 중학교에서 자유학기제가 실시되고 있다(교육부, 2017). 교육과정 편성은 오전에는 국어, 사회, 수학, 과학 등 교과 수업이 주로 이뤄지고, 오후에는 주제 선택 활동, 예술·체육 활동, 동아리 활동, 진로 탐색 활동의 4개 영역으로 특화된 자유학기제 활동 수업이 진행된다. 자유학기제 활동 수업은 한 학기만 운영하는 자유학기의 경우 170시간, 1년 동안 운영하는 자유학년의 경우 221간을 운영하며 학기당 운영시간 및 개설 영역은 학교에서 자율적으로 결정한다.

2025년부터는 [그림1]과 같이 자유학기제의 예술·체육 활동, 동아리 활동과 창의적 체험 활동의 동아리 활동, 학교스포츠클럽 활동과 중복을 최소화하기 위하여 4개 영역을 주제 선택 활동, 진로 탐색 활동 2개 영역으로만 운영한다(교육부, 2022b). 자유학기는 1학년 중 적용 학기를 자율적으로 선택하여 102시간을 운영하며, 진로 탐색 활동의 중요성에 따라 3학년 2학기에 진로연계 학기를 도입한다. 자유학기에서는 주제 선택 활동과 진로 탐색 활동에서 기초소양을 함양할 것을 강조하고 있다.

[그림 Ⅱ-1] 자유학년 활동 수업 예시(교육부, 2022b)

나. 기초소양과 디지털 소양

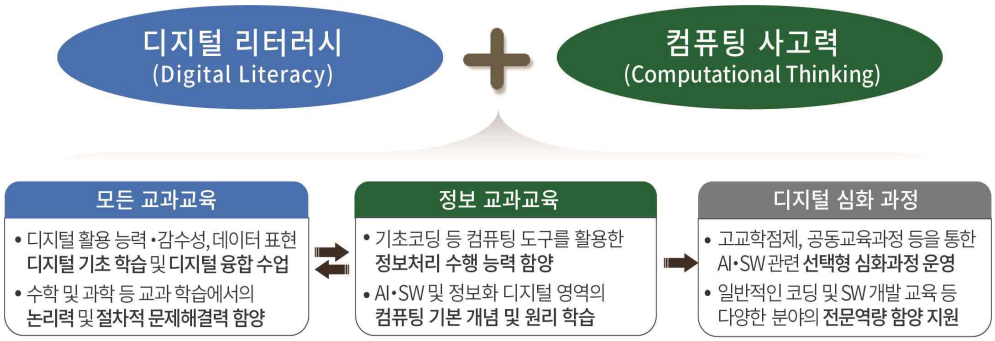
기초소양이란, 여러 교과를 학습하는 데 기반이 되는 역량으로서 미래 사회 변화에 대응할 수 있는 역량이다. 교육부(2022b)는 [그림Ⅱ-2]와 같이 언어 소양, 수리 소양, 디지털 소양 등으로 기초소양으로 강조하고 이러한 기초소양 함양 교육을 총론과 교과에 반영하였다.

기초소양		개념(안)
언어 소양		언어를 중심으로 다양한 기호, 양식, 매체 등을 활용한 텍스트를 대상, 목적, 맥락에 맞게 이해하고, 생산·공유, 사용하여 문제를 해결하고 공동체 구성원과 소통하고 참여하는 능력
수리 소양		다양한 상황에서 수리적 정보와 표현 및 사고 방법을 이해, 해석, 사용하여 문제해결, 추론, 의사소통하는 능력
디지털 소양		디지털 지식과 기술에 대한 이해와 윤리의식을 바탕으로, 정보를 수집·분석하고 비판적으로 이해·평가하여 새로운 정보와 지식을 생산·활용하는 능력

[그림 Ⅱ-2] 기초소양의 개념(교육부, 2022b)

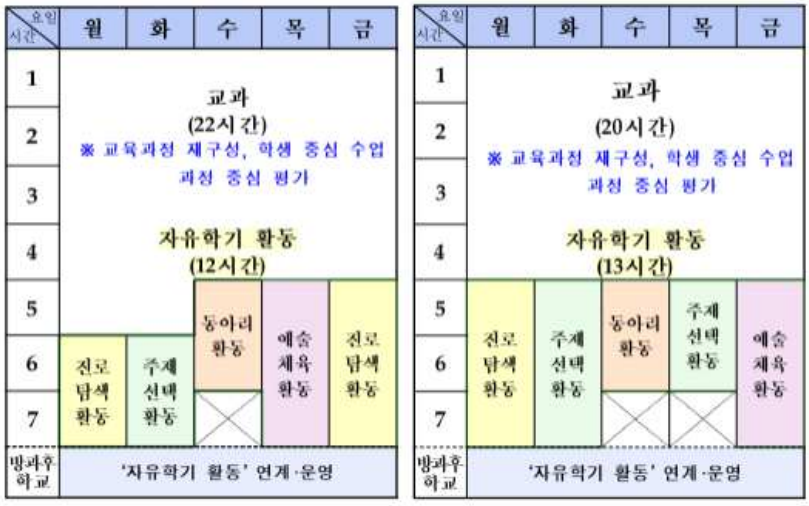
이 중 디지털 소양을 교육부(2022b)는 [그림 Ⅱ-3]과 같이 디지털 리터러시(Digital Literacy)와 컴퓨팅 사고력(Computational Thinking)을 포함하는 역량으로 제시하고 있다. ISTE에서도 2016년도 디지털 소양 개념에 컴퓨팅 사고력을 포함함으로써 컴퓨팅 사고력을 디지털 소양의 하위 개념으로 편성하고 있다(신수범 외, 2017). 컴퓨팅 사고력에 대한 자세한 내용은 Ⅱ-2장의 2.3에서 다루도록 하겠다.

교육부(2022b)에서는 디지털 기초소양 및 컴퓨팅 사고력 함양을 위한 교육과정 구성 방안으로 모든 교과교육과 정보 교과교육을 연계할 것을 제안하고 있다. 이에 본 연구에서는 자유학기에서 기초소양 중 디지털 소양의 컴퓨팅 사고력을 함양할 수 있도록 수학과 정보를 융합한 자유학기제 교육과정을 디자인하였다.



[그림 Ⅱ-3] 디지털 소양 함양을 위한 교육과정 구성 방안(교육부, 2022b)

본 연구에서 디자인한 교육과정을 적용하는 자유학기 활동은 주제 선택 활동으로서 [그림 Ⅱ-4]와 같이 일반적으로 한 프로그램당 일주일에 2~3시간으로 진행되며 총 9차시, 17시간으로 운영되고 있다. 또한, 교과 융합 수업, 학생의 흥미, 관심사에 기반한 프로그램 편성과 자기주도학습 경험 제공 등 학생 참여형 수업을 진행한다. 따라서 교육과정을 자유학기 활동 수업 운영 시수에 맞춰 1차시당 2시간의 수업 분량으로 총 8차시. 16시간에 걸친 교육과정으로 디자인하였다.



[그림 Ⅱ-4] 자유학기 활동 수업 예시(교육부, 2015c)

2. 수학과 및 정보과 교육과정

Ⅰ장에서 언급했듯이 2015 개정 교육과정에서는 중학교에 소프트웨어 교육 중심의 정보 교과를 필수 과목으로 지정하였으며, 2021년 2학기부터 고등학교에 '인공지능 수학' 과목을 선택할 수 있도록 하였다. 이번 장에서는 수학과 교육과정과 정보과 교육과정의 내용 체계 중 공통 요소를 살펴보자고 한다.

2.1. 중학교 수학과 및 정보과 교육과정

중학교 정보과 교육과정(교육부, 2015d)에서는 '문제해결과 프로그래밍' 영역의 내용 체계에 핵심 개념으로 알고리즘, 프로그래밍의 내용 요소로 변수와 연산 및 제어 구조가 등장한다(<표 Ⅱ-1>).

<표 Ⅱ-1> 중학교 정보과 '문제해결과 프로그래밍' 영역 내용 체계

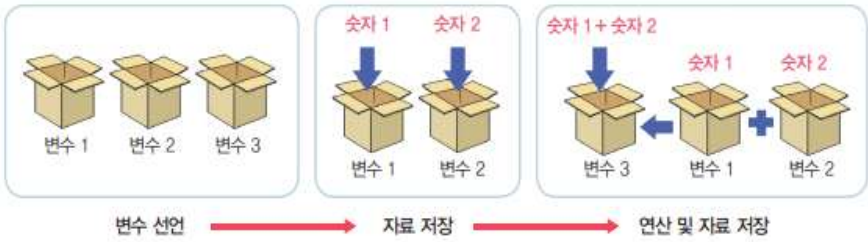
영역	핵심 개념	내용 요소
문제해결과 프로그래밍	추상화	• 문제 이해 • 핵심요소 추출
	알고리즘	• 알고리즘 이해 • 알고리즘 표현
	프로그래밍	• 입력과 출력 • 변수와 연산 • 제어 구조 • 프로그래밍 응용

이 중 변수는 중학교 1학년 수학 교과의 그래프 단원 등 '함수' 영역에서 주요한 수학적 개념으로 중학교 수학과 '함수' 영역의 학습 요소에 좌표, 순서쌍, 좌표평면 등과 함께 변수가 포함되어 있다(<표 Ⅱ-2>).

<표 Ⅱ-2> 중학교 수학과 '함수' 영역 내용 체계

성취기준	[9수03-01] 순서쌍과 좌표를 이해한다. [9수03-02] 다양한 상황을 그래프로 나타내고, 주어진 그래프를 해석할 수 있다. [9수03-03] 정비례, 반비례 관계를 이해하고, 그 관계를 표, 식, 그래프로 나타낼 수 있다. [9수03-04] 함수의 개념을 이해한다.
학습요소	변수, 좌표, 순서쌍, x좌표, y좌표, 원점, 좌표축, x축, y축, 좌표평면, 제1사분면, 제2사분면, 제3사분면, 제4사분면, 그래프, 정비례, 반비례, 함수

이처럼 중학교 정보과 및 수학과 교육과정에는 공통 학습 요소로 '변수'가 포함되어 있다. 정보 교과의 변수와 수학 교과의 변수는 각 교육과정에서 의미하는바 또한 같다. 중학교 정보 교과서(정영식, 2020)에서는 변수를 [그림 Ⅱ-5]와 같이 숫자를 담을 수 있는 상자로 비유하여 설명하고 있는데, 단계를 살펴보면 다음과 같다.

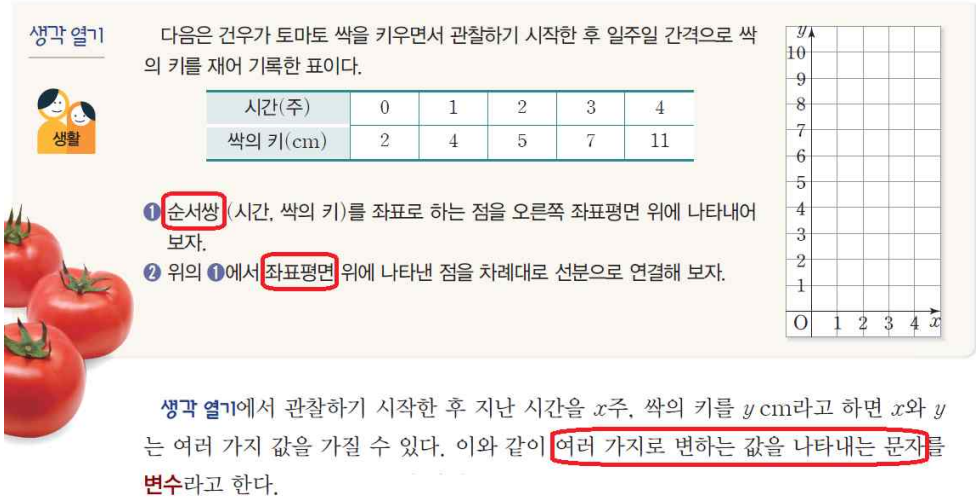


[그림 Ⅱ-5] 중학교 정보 교과에서의 변수(정영식, 2020)

1. 변수 선언 : 빈 상자 3개에 각각 '변수1' '변수2', '변수3'을 붙인다.
2. 자료 저장 : 빈 상자인 '변수1'과 '변수2' 상자에 각각 '숫자1'과 '숫자2'를 넣어 숫자를 저장한다.
3. 연산 및 자료 저장 : '변수1'에 들어 있는 '숫자1'과 '변수2'에 들어 있는 '숫자2'를 새로운 빈 상자 '변수3'에 합쳐서 넣는다.

Papert(1980)는 LOGO 언어에서 학습자의 행동 양식에 기반한 명령어를 강조하였는데, 이렇게 변수를 의미하는 빈 상자에 자료에 해당하는 숫자를 넣어 상자의 값을 변화시키는 과정은 papert의 행동 놀이를 반영한 것이라고 할 수 있다.

중학교 수학 교과서(이준열 외, 2020)에서는 변수를 [그림 Ⅱ-6]과 같이 여러 가지로 변하는 값을 나타내는 문자로 설명하고 있다. 변수를 설명할 때, 순서쌍 및 좌표평면도 함께 다루고 있는데, 수학 교과의 함수 영역에서 변수의 개념을 설명하기 위해서는 순서쌍, 좌표, 좌표평면을 함께 다루어야 함을 알 수 있다.


생각 열기

생활

다음은 건우가 토마토 싹을 키우면서 관찰하기 시작한 후 일주일 간격으로 싹의 키를 재어 기록한 표이다.

시간(주)	0	1	2	3	4
싹의 키(cm)	2	4	5	7	11

❶ 순서쌍 (시간, 싹의 키)를 좌표로 하는 점을 오른쪽 좌표평면 위에 나타내어 보자.

❷ 위의 ❶에서 좌표평면 위에 나타낸 점을 차례대로 선분으로 연결해 보자.

생각 열기에서 관찰하기 시작한 후 지난 시간을 x주, 싹의 키를 y cm라고 하면 x와 y는 여러 가지 값을 가질 수 있다. 이와 같이 여러 가지로 변하는 값을 나타내는 문자를 **변수**라고 한다.

[그림 Ⅱ-6] 중학교 수학 교과에서의 변수(이준열 외, 2020)

2.2. 중학교 정보과 및 고등학교 교육과정

고등학교 교육과정에서 선택 과목으로 도입된 '인공지능 수학' 과목의 '인공지능과 수학' 영역 내용 체계에서는 관련 학습 요소에 진리표와 순서도가 포함되어 있다(<표 Ⅱ-3>).

<표 Ⅱ-3> 고등학교 '인공지능과 수학' 영역 내용 체계

영역	• 인공지능과 수학
일반화된 지식	• 수학은 인공지능의 발전을 이끌어 왔으며, 인공지능 기술 전반에 활용되고 있다.
성취기준	• 인공지능의 발전에 기여한 역사적 사례에서 수학이 어떻게 활용되었는지를 이해한다. • 인공지능에 수학이 활용되는 다양한 예를 찾을 수 있다.
관련 학습 요소	• 진리표 • 순서도

진리표는 5차 교육과정에서 삭제되었으나 진리표 용어만 삭제되었을 뿐 집합은 7차 교육과정까지 <표 Ⅱ-4>와 같이 중학교 1학년 '수와 연산' 영역의 내용 체계에 포함되어 있었으며, 2007 개정 교육과정에서 고등학교 교육과정으로 내용이 모두 이동되었다.

알고리즘과 순서도는 6차 교육과정에 삭제되었으나 <표 Ⅱ-1>의 중학교 정보과의 내용 체계에서도 확인할 수 있듯이 코딩교육에서 핵심 개념이므로 고등학교 수학 교과의 '인공지능과 수학' 영역에서 관련 학습 요소로 '진리표'와 '순서도'가 다시 추가되었다. 이처럼 수학 교과에서는 '진리표'와 '순서도'를 다루고 있으며, 정보 교과에서는 '알고리즘'을 다루고 있으므로 '알고리즘과 순서도'는 수학 교과와 정보 교과의 주요한 공통 학습 요소이다.

<표 Ⅱ-4> 7차 교육과정 수학과
'수와 연산' 영역 내용 체계(교육부, 2006)

학년 / 영역	중학교 1학년	고등학교 1학년
수와 연산	• 집합 • 소인수분해 • 최대공약수, 최소공배수 • 십진법과 이진법 • 정수의 개념과 대소 관계, 사칙계산 • 유리수의 개념과 대소 관계, 사칙계산	• 집합의 연산법칙 • 명제와 조건 • 명제의 역, 이, 대우 • 필요조건과 충분조건 • 실수의 연산 성질, 대소 관계 • 복소수의 뜻과 기본 성질 • 복소수의 사칙계산

이를 바탕으로 시대 변화의 흐름에 맞춰 진리표와 순서도가 추가된 것을 고려하여 '알고리즘과 순서도' 내용을 중학교 1학년 자유학기제 교육과정에 되살릴 필요가 있다. 이에 본 연구에서는 정보 교과의 '순서도'를 수학 교과와 융합하여 교육과정을 디자인하였다.

한편, 2015 개정 교육과정에서는 '기하와 벡터'가 기하'로 과목명이 변경되면서 진로 선택 과목으로 지정되었으며, 대단원 '공간도형과 공간벡터'의 중단원 '공간벡터'가 삭제되었다. 이에 금종혜 외(2018)는 기하는 4차 산업혁명 시대 핵심역량인 창의력, 상상력, 문제해결 능력을 키우고 미래 사회에는 기하의 개념이 있어야 하는 직업군이 더 많이 생길 것으로 예상됨에 따라 학생들이 기하를 배우지 않는다면 우리나라는 미래 사회를 이끌어갈 과학·기술 인재 양성에 어려움이 많을 것으로 우려된다고 하였다. 자유학기제의 목적인 학생들이 진로를 탐색할 수 있도록 공간도형 및 기하와 관련된 내용을 중학교 1학년 자유학기제 교육과정에 반영하고자 한다. 다음 장에서는 공간도형 및 기하를 다룰 수 있는 3차원 좌표계 기반 코딩환경인 터틀크래프트에 대해 논한다.

2.3. 수학적 사고력과 컴퓨팅 사고력

2022 개정 교육과정에서 강조하는 기초소양 중 하나인 디지털 소양에서는 앞의 [그림 Ⅱ-3]과 같이 컴퓨팅 사고력이 포함되어 있다. 이번 장에서는 정보 교과의 컴퓨팅 사고력의 개념 및 구성요소를 바탕으로 수학 교과의 수학적 사고력과 관련성을 살펴보자고 한다.

가. 컴퓨팅 사고력의 개념

Papert(1980)는 코딩환경이 학생들의 강력한 아이디어를 개발하는 도구를 제공하는 학습 환경이라고 주장하며, '가자'와 '돌자'의 몸동작에 기반한 거북 명령 프로그래밍 언어 LOGO를 개발하였다. MIT 대학의 수학자였던 Papert는 컴퓨팅 사고력(CT)[1] 용어를 처음으로 도입하였으며, 컴퓨팅 사고력은 정보사회에 요구되는 새로운 능력으로 인간의 사고와 컴퓨터의 능력을 조합하는 의미에서 강조되고 있다고 하였다(황요한 외, 2016). 이에 우리나라에서도 2018년부터 컴퓨팅 사고력을 기를 수 있도록 소프트웨어교육 중심 교육을 강조하여 앞서 언급했듯이, 중학교에서 '정보 교과'를 필수 과목으로 지정하였으며, 고등학교에서 '인공지능 수학'을 선택 과목으로 도입하였다.

Wing(2006)에 따르면 컴퓨팅 사고력은 컴퓨터 과학자처럼 사고하는 것으로 컴퓨터 과학의 근본적인 개념을 사용하여 문제를 해결하고 시스템을 설계하며 인간의 행동을 이해하는 것을 의미한다. 또한, Wing은 컴퓨팅 사고력을 컴퓨터(인간 또는 기계)가 효과적으로 수행할 수 있는 방식으로 문제를 공식화하고 해법을 표현하는 것과 관련된 사고 과정으로 정의하였다(Wing, 2017, p. 8; 신동조, 2019에서 재인용).

---

1) Computational Thinking은 '컴퓨팅 사고력', '컴퓨터적 사고', '컴퓨팅적 사고' 등으로 다양하게 번역되고 있으나, 과학교육 분야에서는 Ministry of Education과 한국과학창의재단을 중심으로 '컴퓨팅 사고력'으로 번역하고 있으므로, 본 연구에서도 '컴퓨팅 사고력'을 사용하도록 하였다.

나. 컴퓨팅 사고력의 구성요소

International Society for Technology in Education & Computer Science Teachers Association(ISTE & CSTA, 2011)은 컴퓨팅 사고력을 여러 분야에서 문제를 해결하는데 필요한 능력으로 보고, 컴퓨팅 사고력의 구성요소를 자료 수집, 자료 분석, 자료 표현, 문제 분해, 추상화, 알고리즘 및 절차화, 자동화, 검증, 일반화 등으로 제시하였다. 구성요소 각각의 정의를 정리하면 <표 Ⅱ-5>와 같다.

<표 Ⅱ-5> 컴퓨팅 사고력의 구성요소(ISTE & CSTA, 2011)

구성요소	정의
자료 수집 (Data Collection)	적절한 정보를 수집하는 과정
자료 분석 (Data Analysis)	자료를 이해하고 패턴을 찾아 결론을 파악하는 과정
자료 표현 (Data Representation)	자료를 그래프, 표, 단어, 이미지로 적절하게 표현하고 구성하는 과정
문제 분해 (Problem Decomposition)	문제를 해결하기 위해 문제를 나누어 분석하는 과정
추상화 (Abstraction)	문제의 복잡도를 줄이기 위해 기본 주요 아이디어를 설정하는 과정
알고리즘과 절차화 (Algorithms & Procedures)	문제를 해결하기 위한 과정을 순서적 단계로 표현하는 과정
자동화 (Automation)	컴퓨터나 기계가 반복적인 작업을 하도록 하는 과정
검증 (Simulation)	모의 실험하는 단계
일반화 (Parallelization)	목표를 달성하기 위한 작업을 동시에 수행하도록 리소스를 구성하는 단계

다. 컴퓨팅 사고력의 추상화와 자동화

Wing(2008)은 컴퓨팅 사고력의 핵심은 추상화된 개념을 자동화의 도구를 이용하여 문제해결을 수행하는 것이라고 하였다. Wing은 추상화는 인간이 할 수 있는 정신적인 도구라고 말하였으며, 추상화된 개념은 컴퓨터에 의해 자동화되어 반복적으로 수행될 수 있다고 하였다(한치근, 2018에서 재인용). 이처럼 Wing은 컴퓨팅 사고력의 구성요소 중 핵심 구성요소로 추상화와 자동화를 제시하였으며, 컴퓨팅 사고력의 구성요소에서 추상화와 자동화는 따로 분리하여 생각하는 것이 아니라 함께 일어나는 과정임을 알 수 있다.

앞서 ISTE & CSTA(2011)는 컴퓨팅 사고력의 중요한 특성인 추상화 과정을 주요 아이디어를 정의하기 위한 복잡성을 줄이는 과정이라고 정의하였고, 컴퓨팅 사고력의 자동화 과정을 컴퓨터나 기계가 반복적인 작업을 수행하는 과정이라고 정의하였다.

[그림 Ⅱ-7] 중학교 정보 교과서에서 추상화와 자동화(정영식, 2020)

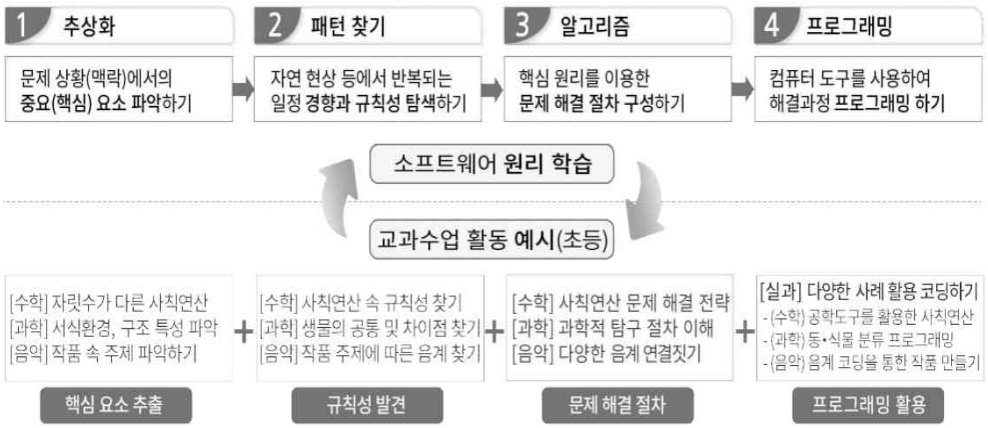
구체적으로 중학교 정보 교과서(정영식, 2020)에서는 [그림 Ⅱ-7]과 같이 사용한 블록의 개수를 11개에서 6개로 줄이는 과정이 있다. 이는 반복되는 부분을 주요 아이디어로 정의하여 복잡성을 줄이며 컴퓨터가 반복해서 실행하는 과정이므로 바로 추상화 및 자동화 과정이라고 할 수 있다. [그림 Ⅱ-7]에서 이동 방향으로 100만큼 움직이고 방향을 90°만큼 회전하는 명령을 주요 아이디어로 정의하는 과정이 바로 추상화 과정이며 이를 컴퓨터가 기계적으로 4번 반복하는 과정이 자동화 과정이다.

이처럼 2015 개정 교육과정의 중학교 정보 교과에서는 추상화와 자동화를 핵심 개념으로 다루고 있으며, 추상화와 자동화가 컴퓨팅 사고력을 신장시키는 중요한 특성이라고 할 수 있다. 이러한 추상화 과정은 수학 교과의 중요한 특성이기도 하다. 제7차 수학과 교육과정에서는 수학적 지식의 특성 요인 중 하나로 추상화를 제시하였다. 따라서, 수학 교과의 수학적 사고력과 정보 교과의 컴퓨팅 사고력은 추상화를 공통적인 특성으로 지니고 있다.

이미 많은 선행연구에서는 컴퓨팅 사고력과 수학적 사고의 공통 요소 중 하나로 추상화 과정을 제시하고 있다. 컴퓨팅 사고력의 구성요소 중 장경윤(2017)은 컴퓨팅 사고력과 수학교육의 공통 요소로 알고리즘, 추상화, 분해, 패턴 인식을, 유가영(2017)은 컴퓨팅 사고력과 수학적 사고의 공통 요소로 알고리즘과 절차, 추상화, 문제 분해를 추출하였으며, 남충노, 김종우(2011)는 컴퓨팅 사고력과 수학적 사고의 공통 요소로 알고리즘적 사고, 논리적 사고, 비판적 사고, 재귀적 사고를 추출하였다. National Research Council(NRC, 2010)에 따르면, 컴퓨팅 사고력과 수학적 사고는 추상화를 통해 단순화된 모델에 관한 추론을 기반으로 한다는 점에서 공통점을 가지며, 고유의 언어(프로그래밍 언어와 수학기호)를 통해 대상을 표현한다는 점에서 비슷하다고 하였다.

앞서 언급한 2022 개정 교육과정에서는 초·중·고등학교 모두 학교급별 발달 단계에 따라 모든 교과 교육을 통해 디지털 기초소양 함양할 수 있도록 내용 기준을 개발할 것을 제안하고 있다. 디지털 기초소양에는 컴

퓨팅 사고력이 포함되므로 결국 교육부는 컴퓨팅 사고력을 신장시키기 위해 정보 교과와 연계한 융합 교육과정을 개발할 것을 지향한다. 컴퓨팅 사고력의 핵심 구성요소는 추상화이고, 수학적 사고력의 주요 특성은 추상화 과정이며, 정보 교과와 수학 교과의 공통적인 학습 요소로 알고리즘과 순서도가 있다. 교육부(2022b)에서는 [그림 Ⅱ-8]과 같이 정보 교과의 추상화, 패턴 찾기, 알고리즘, 프로그래밍의 학습 단계를 수학, 과학, 음악 등 다른 교과와 연계한 교과수업 활동 예시를 제안하고 있다.



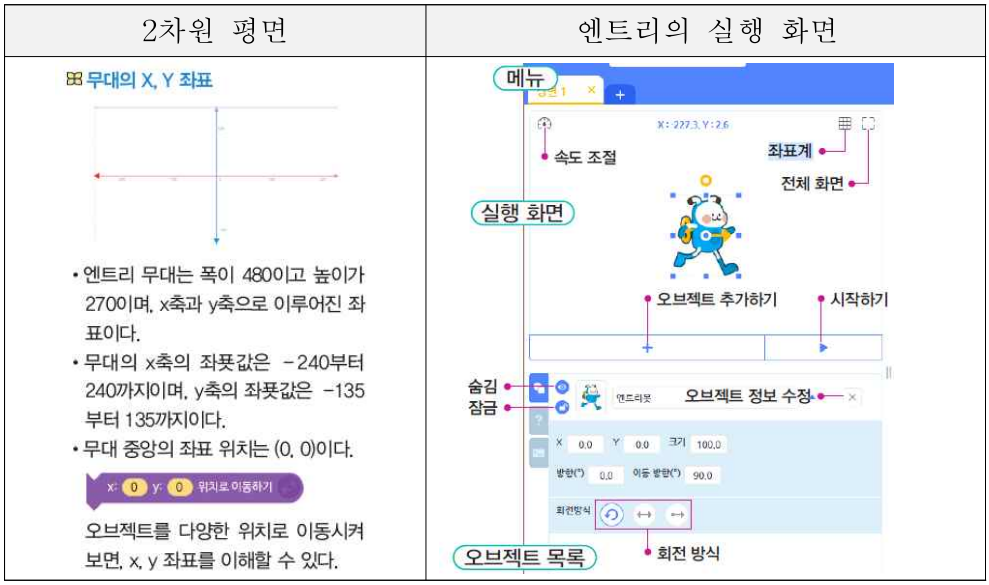
[그림 Ⅱ-8] 정보 교육과정과 연계 활동 예시(교육부, 2022b)

이에 본 연구에서는 수학 교과와 정보 교과의 공통 학습 요소를 바탕으로 추상화 과정을 통해 컴퓨팅 사고력을 신장시키기 위하여 수학적 요소를 융합한 코딩환경을 활용하고자 한다.

3. 터틀크래프트

3.1. 3차원 좌표공간

중학교 정보 교과서(정영식, 2020)에서는 [그림 Ⅱ-9]와 같이 코드의 실행 결과가 2차원 좌표평면으로 나타나는 엔트리(Entry)나 스크래치(Scratch) 소프트웨어를 사용하고 있다. 우리가 생활하는 공간은 3차원으로 2차원 평면을 넘어 3차원 공간을 익히기 위해서는 3차원 좌표 학습을 다룰 필요가 있다. 이에 한국과학창의재단 및 교육부에서는 3차원 좌표 공간을 활용하는 소프트웨어인 알지오매스(Algeomath)를 개발하였다.

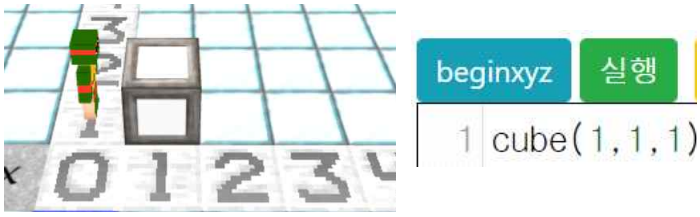


[그림 Ⅱ-9] 엔트리의 코드 실행 결과 화면

알지오매스는 한국과학창의재단이 교육부, 17개 시·도 교육청의 지원을 받아 함께 개발한 초·중·고등학교 수학 실험탐구용 및 도형 학습용 소프트웨어로 2018년 11월 7일부터 누구나 무료로 사용할 수 있도록 정

식 운영을 시작하였다. 정부는 국가 예산에 의해 개발된 공학적 도구로 인해 수학 교과서에 알지오매스라는 명칭을 명시하는 것뿐만 아니라 교수학습, 평가 등에서 적극적으로 활용됨으로써 수학교육의 긍정적인 변화를 선도할 것이라고 기대하고 있다(이환철, 2019). 알지오매스는 타 소프트웨어 교구와는 다르게 3차원 좌표기반, 행동 기반 특징을 가진 소프트웨어이다. 본 연구에서 다루는 터틀크래프트(Turtlecraft)는 알지오매스와 마찬가지로 다양한 입체 구조물을 만들 수 있는 3차원 좌표계 기반 코딩환경으로 행동 기반 특징을 지니고 있다.

Papert(1980)는 '가자, 돌자'와 같은 행동 기반의 LOGO 언어와 학생들이 능동적으로 탐구하며 학습할 수 있는 컴퓨터 공간인 마이크로월드(Microworld)를 제안하였다. 마이크로월드의 대표적인 예로는 Papert의 LOGO 언어를 계승하여 개발된 코딩환경인 Javamal(조한혁 외, 2016)이 있다. 터틀크래프트는 Javamal에 3차원 좌표공간을 도입한 3차원 좌표계 기반 코딩환경으로 자유로운 조작 활동을 할 수 있는 마이크로월드이다. 터틀크래프트에서는 [그림 Ⅱ-10]과 같이 3차원 좌표공간에서 순서쌍으로 나타낸 좌표를 이용하여 쌓기나무를 쌓을 수 있다. 따라서 앞서 <표 Ⅱ-2>에서 살펴본 변수, 순서쌍, 좌표평면 등의 수학적 개념을 코딩과 융합하여 학습할 수 있다.



[그림 Ⅱ-10] 터틀크래프트의 3차원 좌표공간

쌓기나무로 다양한 구조물을 표현하는 LEGO 만들기 활동은 창의성 신장에 효과적이며, 공간 추론 능력, 공간 방향화 능력 등의 공간 능력의 발달에도 도움이 된다(이지윤, 2010; Brosnan, 1998). 터틀크래프트에서

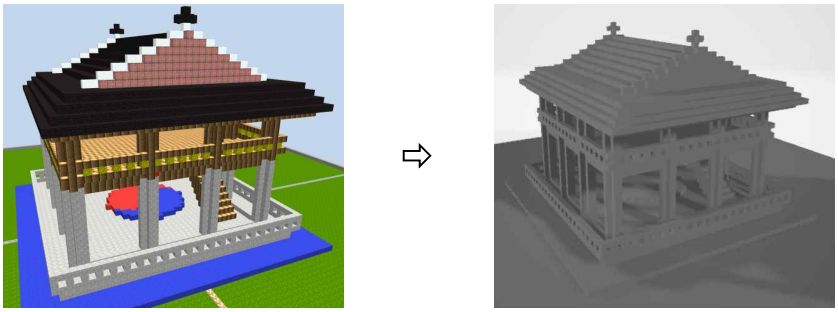
는 3차원 좌표계를 기반으로 다양한 입체도형을 표현할 수 있어 창의성 및 공간 추론 능력의 발달에 도움을 줄 수 있다(정인우, 2020). 공간도형은 고등학교 교육과정에서도 다루는 내용으로 초등학교부터 중·고등학교 이상 높은 수준의 수학적 개념에 대한 탐구까지 나아갈 수 있다는 점에서 다른 마이크로월드와 차별화되는 장점을 가지고 있다(정인우, 조한혁, 2020).

이러한 터틀크래프트는 <그림 Ⅱ-11>과 같이 세계적으로 인기를 끌고 있는 마인크래프트(Minecraft) 게임과 유사한 쌓기나무 기반의 구성 활동과 코딩을 지원한다. 마인크래프트 게임은 초등학생들이 가장 많이 다루는 게임 중 하나로 우리나라뿐만 아니라 전 세계의 어린이들이 서로 경험을 공유하고 엄청난 규모의 어린이들이 디지털 제작에 참여하였다(이명숙, 2019). 터틀크래프트는 이러한 마인크래프트 게임에 공간좌표를 도입하여 학생들에게 코딩으로 수학을 배울 수 있도록 만든 소프트웨어 교구이다. 터틀크래프트는 다른 소프트웨어 교구와는 다르게 마인크래프트를 많이 접한 중학생들에게 매우 익숙한 환경일 수 있다. 이처럼 터틀크래프트는 마인크래프트 게임을 하듯이 접하기 때문에 학생들에게 흥미를 불러일으킬 수 있으며 3차원 좌표공간에서 자신만의 건축물을 제작한다는 목표로 동기를 유발할 수 있다.

<그림 Ⅱ-11> 마인크래프트와 터틀크래프트

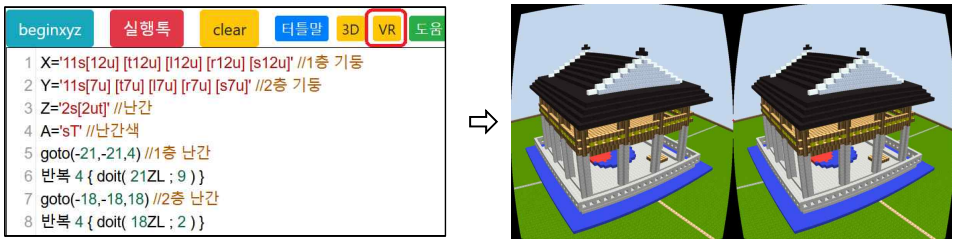
마인크래프트	터틀크래프트
	

또한, 터틀크래프트에서는 3차원 좌표공간에서 제작한 산출물을 3D 프린터로 출력하고, VR기기를 이용하여 3D VR 게임을 다룰 수 있는 환경[2]을 제공한다. [그림 Ⅱ-12]는 3차원 좌표공간에서 입체 구조물의 코드를 OBJ 파일로 변환하여 3D 프린트로 출력하는 과정이다.



[그림 Ⅱ-12] 터틀크래프트를 이용한 3D 프린트 제작

이러한 OBJ 파일은 [그림 Ⅱ-13]과 같이 명령어 창에서 경회루를 만드는 명령어를 입력한 후 3D 버튼을 누르면 생성된다. 마찬가지로 VR 버튼을 누르면 VR기기로 볼 수 있는 화면이 생성된다. 이러한 3D 프린팅과 3D VR 게임 등은 학생들이 꿈과 끼를 찾을 수 있도록 하는 학생참여형 수업, 체험 활동이 가능한 교육과정이라는 점에서 자유학기제의 취지에 맞는 교육과정이라고 할 수 있다.



[그림 Ⅱ-13] 터틀크래프트를 이용한 3D VR 게임 체험

2) 3D 프린팅과 VR 게임이 가능한 터틀크래프트 환경 : hanaone.com

3.2. 텍스트 코딩과 블록 코딩

교육용 프로그래밍 언어는 교육을 목적으로 학습자의 인지 발달 단계를 고려하여 개발된 프로그래밍 언어들을 의미한다(주민정, 2021).

<표 Ⅱ-6> 블록 코딩과 텍스트 코딩

종류	특징
블록 코딩	• 텍스트 명령어 대신 레고 블록과 같은 조립 방식으로 구현 • 주로 스토리, 게임, 애니메이션, 디지털 아트 등의 개발에 사용됨 • 예제 언어 : 스크래치, 엔트리, 앱 인벤터, 코두, 앨리스 등
텍스트 코딩	• 전통적인 프로그래밍 언어에서 교육적으로 사용할 내용만 추출하여 적용 • 알고리즘, 절차적 사고, 추상화, 재사용, 디버깅, 모듈화 등 • 예제 언어 : 파이썬, 자바스크립트, PHP, 베이직 등

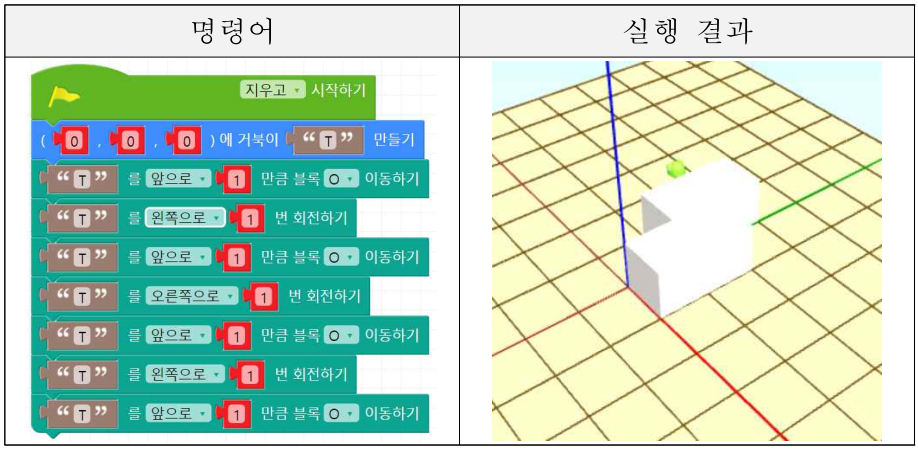
교육용 프로그래밍 언어에는 <표 Ⅱ-6>과 같이 블록(Block) 기반 프로그래밍 언어와 텍스트(Text) 기반 프로그래밍 언어가 있으며, 블록 기반 프로그래밍 언어를 블록 코딩, 텍스트 기반 프로그래밍 언어를 텍스트 코딩이라 부른다. 블록 코딩은 텍스트 코딩을 초등학생 또는 중학생이 초기에 코딩 명령어를 도입할 때 명령어를 쉽게 익히기 위해 개발된 교육용 프로그래밍 언어이다. 블록 코딩에는 앞서 언급한 엔트리와 스크래치가 대표적으로 사용되고 있으며, 텍스트 코딩에는 파이썬(Python)과 자바스크립트(Javascript)가 대표적으로 사용되고 있다. 본 연구에서 다루는 터틀크래프트는 텍스트 기반 프로그래밍 언어로 자바스크립트 언어의 모든 명령이 여기에서 작동된다.

이러한 블록 코딩과 텍스트 코딩은 각각의 특징에 따라 교육적으로 활용하는 데 장단점을 지니고 있다. 이를 정리하면 <표 Ⅱ-7>과 같다.

<표 Ⅱ-7> 블록 코딩과 텍스트 코딩의 장단점

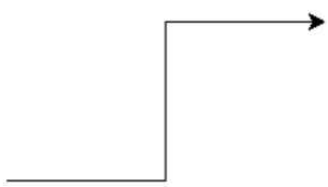
종류	장점	단점
블록 코딩	• 초기에 명령어 쉽게 익히는 것이 가능 • 오타로 인한 명령어 미실행 가능성 거의 없음	• 최종적으로 배워야 할 프로그래밍 언어가 아님 • 명령어가 규격화되어 있어 융통성이 낮음
텍스트 코딩	• 오타로 인한 명령어 미실행 가능성이 큼 • 명령어를 알고 있어야 명령어 작성이 가능	• 최종적으로 배워야 할 프로그래밍 언어에 해당 • 명령어를 활용할 수 있는 융통성이 높음

블록 코딩은 [그림 Ⅱ-14]와 같이 블록 안에 명령어가 있으므로 학생이 명령어를 직접 입력해야 할 필요가 없다. 학생들이 처음 명령어를 접하는 경우 명령어가 이미 제시되어 있으므로 쉽게 익히는 것이 가능하며, 직접 명령어를 입력할 필요가 없어 오타로 인해 명령어가 실행되지 않는 경우가 거의 없어 초등학생 또는 중학생에게 코딩교육을 도입하기에 적합한 프로그래밍 언어이다.



[그림 Ⅱ-14] 알지오매스의 블록 코딩

반면에 텍스트 코딩은 [그림 Ⅱ-15]와 같이 명령어를 직접 입력해야 하므로 명령어를 정확히 입력하지 않아 오타가 있는 경우 실행되지 않는 불편함이 있다. 이처럼 명령어를 초기에 도입하기에는 텍스트 코딩과 비교해 블록 코딩이 더 다루기 쉬운 특징이 있다.

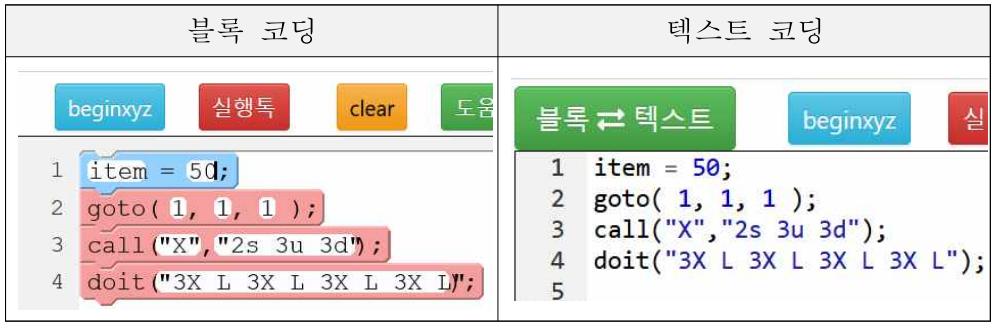
명령어	실행 결과
import turtle t=turtle.Turtle() t.forward(100) t.left(90) t.forward(100) t.right(90) t.forward(100)	

[그림 Ⅱ-15] 파이썬의 텍스트 코딩

그렇다면 최종적으로 다루어야 할 프로그래밍 언어는 무엇일까? 고등학교 정보과 교육과정(교육부, 2015b)에서는 '[12정보04-01] 텍스트 기반 프로그래밍 언어의 개발 환경 및 특성을 이해한다'와 같이 텍스트 기반 프로그래밍 언어를 성취기준으로 삼고 있으므로 고등학교에서는 결국 텍스트 기반 프로그램을 다뤄야 한다. 따라서 코딩교육에서는 초기에 도입할 때 블록 코딩을 활용하더라도 최종적으로는 블록 코딩에서 텍스트 코딩으로의 전환을 요구한다.

또한, 안창호 외(2018)는 현실에서는 텍스트 기반 프로그래밍의 강력함과 유연성을 대체할 수 없다고 하였다. 텍스트 코딩은 명령어를 효율적으로 짧게 줄이는 것이 가능하지만, 블록 코딩은 명령어가 블록으로 규격화되어 있으므로 명령어를 수정하는 데 한계가 있어 명령어를 효율적으로 짧게 구성하기에 어려움이 있다. 특히, 추상화 과정은 명령어의 복잡성을 줄이는 과정이며 텍스트 코딩은 명령어를 융통성 있게 입력할 수 있다는 점에서 텍스트 코딩이 블록 코딩과 비교해 추상화 과정을 더 효과적으로 나타낼 수 있다.

이에 본 연구에서 활용하는 코딩환경인 터틀크래프트에서는 텍스트 코딩과 블록 코딩의 장점을 모두 활용하기 위해 블록 코딩과 텍스트 코딩을 변환할 수 있는 기능을 추가하였다. [그림 Ⅱ-16]과 같이 블록 코딩으로 명령어를 구성한 후, 블록과 텍스트를 전환하는 버튼을 누르면 블록 코딩이 텍스트 코딩으로 변환된다. 블록 코딩을 이용하여 초기에 명령어를 쉽게 도입할 수 있을 뿐만 아니라 블록 코딩으로 명령어를 구성함으로써 오타의 가능성을 줄이고, 텍스트 코딩으로 명령어를 유연하게 수정하여 명령어를 효과적으로 짧게 나타낼 수 있다.



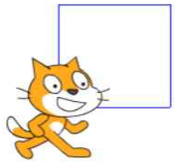
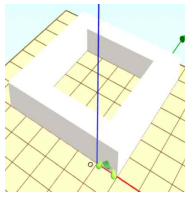
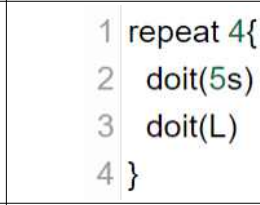
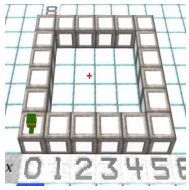
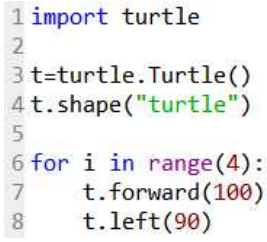
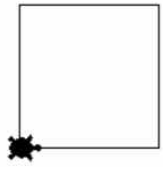
[그림 Ⅱ-16] 터틀크래프트의 블록 코딩과 텍스트 코딩

한편 알지오매스와 터틀크래프트는 모두 3차원 좌표기반이지만, 알지오매스는 블록 코딩, 터틀크래프트는 블록 코딩으로 명령어를 도입하여 최종적으로는 텍스트 코딩을 활용한다. 앞서 언급한 스크래치, 알지오매스, 터틀크래프트, 파이썬의 특징을 정리하면 <표 Ⅱ-8>과 같으며, 크게 다음과 같은 2가지 특징으로 구분할 수 있다.

먼저 코드를 블록 코딩과 텍스트 코딩으로 구분하면, 블록 코딩에는 스크래치와 알지오매스, 텍스트 코딩에는 터틀크래프트와 파이썬이 속한다. 다음으로, 코드의 실행 결과를 2차원 평면과 3차원 공간으로 구분하면, 2차원 평면에서 코드 실행 결과가 나타나는 프로그램은 스크래치와 파이썬이며, 3차원 공간에서 코드 실행 결과가 나타나는 프로그램은 터틀크래프트와 파이썬이 있다. 이처럼 터틀크래프트는 코드의 실행 결과

가 3차원 좌표공간에 나타나면서 최종적으로 다루어야 할 텍스트 기반 프로그래밍 언어이다. 따라서, 본 연구에서는 블록 코딩과 텍스트 코딩의 변환이 가능하면서 3차원 좌표계 기반 코딩환경인 터틀크래프트를 활용한 자유학기제 교육과정을 디자인하고자 한다.

<표 Ⅱ-8> 스크래치, 알지오매스, 터틀크래프트, 파이썬의 코드 및 코드 실행 결과의 비교

프로그램	코드	코드 실행 결과	특징
스크래치			• 블록 코딩 • 2차원 평면
알지오매스			• 블록 코딩 • 3차원 공간
터틀크래프트			• 텍스트 코딩 • 3차원 공간
파이썬			• 텍스트 코딩 • 2차원 평면

4. 최소 코딩게임

4.1. 최소 코딩게임의 개념

코딩교육에서의 핵심 목표는 컴퓨팅 사고력을 신장시키는 것이며, 컴퓨팅 사고력의 주요한 특성은 추상화 및 자동화 과정이다. 최소 코딩게임(Minimum coding game) 교수전략은 [그림 Ⅱ-7]과 같이 블록의 개수를 줄이는 과정인 추상화 및 자동화 과정을 학생이 효과적으로 학습하기 위해 도입한 교수전략이다. 최소 코딩게임은 이미 여러 선행연구에서 효과적인 교수전략으로 사용되었으며, 최소 표현 게임, 최소코드 게임 등의 용어로 불리고 있다.

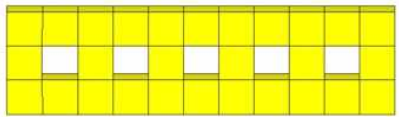
정진환(2015)은 '최소 표현 게임'을 학생들에게 패턴을 발견하고 표현하는 것을 유도하는 활동으로 정의하였으며, 최소 표현 게임을 통해 학생이 대상을 추상화한다고 설명하였다. 정진환, 조한혁(2020)은 '최소코드 게임'을 가장 적은 코드를 사용하여 대상을 구성하는 게임이라고 정의하였으며, 산술적 코딩에서 대수적 코딩으로 발전되는 교육 내용을 설계하고 대수적 코딩으로 나아가도록 하는 최소코드 지도전략을 제안하였다. 최소코드 게임은 다음과 같은 두 가지 특징을 지닌다.

1. 주어진 입체 구조물을 만드는 명령어를 가장 짧게 최소(Minimum)로 표현하도록 유도한다.
2. 주어진 입체 구조물을 만드는 문제를 명령어를 수치화하는 게임(game) 형태로 제시한다.

예로 [그림 Ⅱ-7]에서는 블록의 개수를 11개에서 6개로 줄이고 있다. 최소코드 게임을 활용하면 최소로 블록의 개수를 사용하자는 조건을 붙여 문제를 제시하며, 블록의 개수를 이용하여 명령어를 수치화한다.

4.2. 최소 코딩게임의 원리

최소 코딩게임은 '가장 짧게' 명령어를 코딩하는 게임 형태의 문제이다. 학생에게 입체 구조물을 제시하고 학생이 그 구조물을 만드는 명령어를 코딩하였을 때, 가장 짧게 코딩하였는지 판단하기 위해서는 판단 기준이 필요하다.



[그림 Ⅱ-17] 최소코드 게임의 예시(정진환, 조한혁, 2020)

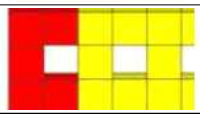
정진환, 조한혁(2020)은 최소코드 게임의 예시로 [그림 Ⅱ-17]'과 같은 입체 구조물을 제시하며, 판단 기준으로 코드의 개수를 두고 코드의 개수는 글자의 개수로 세도록 하였다. 이때, 명령어를 짧게 쓰는 것이 목적이므로 명령어를 짧게 코딩하기 위해 사용된 'X=', '[', ']'와 따옴표의 개수는 세지 않는다. 명령어는 하나의 단어이고 단어의 길이에 따라 코드의 개수를 판단해서는 안 되므로 명령어는 1개로 센다. 최소코드 게임의 규칙을 정리하면 다음과 같다.

1. 코드를 가장 짧게 쓸 것
2. 코드의 개수는 글자의 개수로 센다.
3. 숫자 1개, 문자 1개당 글자의 개수 1개로 센다.
4. X=, [,], 따옴표는 개수로 세지 않는다.
5. do 명령어는 1개로 센다.

정진환, 조한혁(2020)에 의하면 [그림 Ⅱ-17]에서 제시한 입체 구조물에 대한 학생들의 응답을 대표적으로 <표 Ⅱ-9>로 나타낼 수 있다. 학생이 명령을 가장 짧게 코딩하기 위해서는 추상화가 필수적이다. 추상화

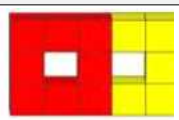
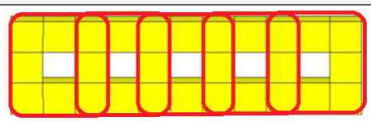
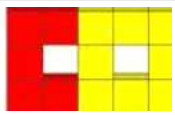
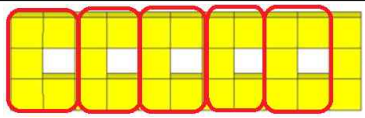
는 주요 아이디어인 반복되는 부분을 정하여 복잡성을 줄이는 과정이므로 반복되는 부분을 어떻게 인지하느냐에 따라 코드의 개수가 달라진다.

<표 Ⅱ-9> 최소코드 게임의 학생 답안 예시

	유형	코드	코드의 개수
①		X = 's [2u 2s 2d] s ' do 5X	11
②		X = '2s [2u 2t]' do s [2u] 5X	12

유형 ①에서는 'ㅁ' 모양을 반복되는 부분으로 생각하고 명령어를 코딩하였으며, 유형 ②에서는 'ㄷ' 모양을 반복되는 부분으로 생각하고 명령어를 코딩하였다. 코드의 개수가 유형 ①보다 유형 ②가 더 적으므로 유형 ①이 명령어가 더 짧다. 이는 반복되는 부분으로 제시된 입체 구조물을 모두 표현할 수 있느냐에 따라 코드의 개수가 달라진다. <표 Ⅱ-10>에서 유형 ①과 유형 ② 모두 반복 횟수는 5번으로 같지만 'ㅁ' 모양으로는 제시된 입체 구조물을 모두 나타낼 수 있는 반면에 'ㄷ' 모양으로는 제시된 입체 구조물을 모두 나타낼 수 없으므로 명령어가 더 추가되어 코드가 길어진다. 따라서 최소코드 게임에서는 반복되는 부분을 어떻게 인지하고 설정하느냐에 따라 명령어의 길이가 달라질 수 있다.

<표 Ⅱ-10> 최소코드 게임의 학생 답안 예시 분석

	유형	반복되는 부분	반복 횟수
①			5
②			5

4.3. 문제의 정의와 좋은 문제의 요소

가. 문제의 정의

Kriulik & Rudnick(1987)은 문제란 '개인이나 개인의 집단이 직면하고, 해를 요구하며, 개인이 해를 얻기 위한 어떤 외견상의 명확한 수단이나 길이 존재하지 않는 상황'이라고 정의하였다. Polya는 문제를 가지고 있다는 것은 '분명하게 이해되었지만, 즉각적으로 도달할 수 없는 목표에 이르기에 적당한 행동을 의식적으로 찾고 있는 것'이라고 말하였다(김윤민, 2005). Kriulik & Rudnick(1987)은 문제해결이란 '하나의 과정이고, 그것은 한 개인이 낯선 문제 상황이 요구하고 있는 것을 만족시키기 위하여 이전에 획득한 지식, 기능, 그리고 이해를 이용하기 위한 수단'이라고 정의하고 학생은 그가 이전에 배웠던 것을 종합해야 하고 그것을 새롭고 다른 상황에 적용해야 한다고 주장하고 있다.

최소 코딩게임 형태의 문제는 학생들이 그동안 간단한 입체 구조물을 만들면서 배운 명령어를 종합하여 새로운 입체 구조물을 만들 때 명령어를 짧게 쓰는 과정을 유도하므로 문제해결 과정이라고 할 수 있다.

나. 좋은 문제의 요소

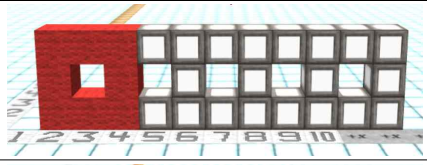
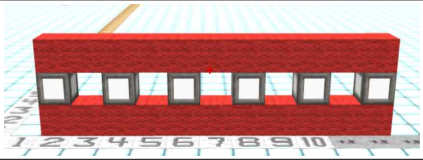
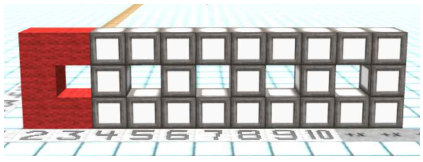
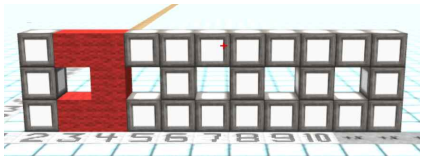
Kriulik & Rudnick(1987)은 다음과 같은 특성의 일부 또는 전부를 포함하는 것을 좋은 문제라 제안하였다.

1) 문제에 대한 해는 명확한 수학적 개념의 이해 또는 수학적 기능의 사용을 포함하여야 한다.
2) 문제의 해는 일반화를 이끌어야 한다.
3) 문제는 확장의 기회를 줄 수 있는 개방형이어야 한다.
4) 문제 곧 그 자체가 다양한 해답을 지녀야 한다.
5) 문제는 학생들에게 흥미와 도전 의식을 일으켜 주어야 한다.

최소 코딩게임 교수전략을 적용하기 위해 터틀크래프트를 활용하여 제시하는 게임 형태의 문제는 이러한 좋은 문제의 특성을 포함하도록 구성되었다. 이를 정리하면 다음과 같다.

1) 명령어를 짧게 쓰기 위해서 반복되는 부분을 치환 문자를 이용하여 치환하므로 수학적 개념을 이해하면서도 치환 문자라는 수학적 기능을 사용하도록 유도한다.
2) 명령어를 일반화하면 명령어의 개수는 변하지 않으면서 계수의 변화를 통한 확장이 가능하다.
3) 반복되는 부분을 구성하는 방법에 따라 주어진 입체 구조물을 만들 수 있는 다양한 명령어가 존재한다(<표 Ⅱ-11>).
4) 게임 형태 문제를 통해 학생들에게 흥미와 도전 의식을 일으킨다.

<표 Ⅱ-11> 최소 코딩게임 문제의 예시

반복되는 부분	코드
	X='2s [2u 2t 2d]' goto(2, 1, 1); doit(5X)
	X='11s' goto(1, 1, 1); doit(X) goto(1, 1, 3); doit(X) …
	X='2s [t 2u s]' goto(1, 1, 1); doit(5X) doit(s 2u)
	X='2s [2u t]' goto(1, 1, 1); doit(s [2u]) doit(5X)

5. 학습 동기

5.1. 학습 동기의 개념

동기(motivation)는 '어떠한 행동을 유발시키고, 그 행동을 유지하고, 목표를 향해 나아가도록 하는 힘'으로 정의되며(Pintrich & Schunk, 2002), 학습 동기란 배우고자 하는 의욕을 나타낸다.

5.2. 학습 동기의 요소

Pintrich & Schunk(2002)는 학습 동기를 내재적 동기와 외재적 동기로 나누었으며, 내재적 동기는 공부하는 것 그 자체를 목적으로 하는 동기이며, 외재적 동기는 공부가 목적이 아니라 공부하는 것이 최종 목표의 수단으로 사용되는 동기로 도구적 동기라고 하였다. Coroll(1963)과 Ryan(1992)은 학습 동기의 구분을 내적 동기(intrinsic motivation)와 외적 동기(extrinsic motivation)로 구분하였다. 내적 동기는 학습자가 어떤 외부로부터의 보상을 생각하지 않고 학습 목표를 세워 자신의 방법을 설정하고 추진하는 자신의 의지에서 나오는 동기를 말한다. 외적 동기는 학습자가 자신의 목표를 달성하기 위해서 스스로 노력하기보다는 외부로부터 오는 보상을 바라며 학습하고자 하는 욕구를 가질 때 이를 지칭한다(곽아영, 2011).

내적 학습 동기가 학습 목표 자체가 목적으로 학습하는 것 그 자체가 목적이므로 내재적 동기라고 할 수 있으며, 외적 학습 동기는 보상이라는 최종 목표를 위해 학습하고자 하는 욕구를 지니어 학습을 수단으로 사용하므로 외재적 동기라고 할 수 있다.

가. 내적 학습 동기

내적 학습 동기는 경제적 보상을 바라지 않고 칭찬, 좋은 성적, 경쟁 욕구와 같은 심리적 요인에 의해 발생하기 동기이다. 높은 수준의 내적 학습 동기를 가진 학생들이 낮은 수준의 내적 학습 동기를 갖거나 외적 학습 동기를 가진 학생들에 비해 더 높은 학업적 도전에 직면하기를 좋아하며, 학습 내용에 호기심과 흥미를 보이는 것으로 나타났다(Newman, 1990; 박현정, 2008에서 재인용).

본 연구에서는 높은 수준의 내적 학습 동기를 지니고 코딩교육의 학습 수준이 높은 학생들을 위해 도전 의식을 불러일으키도록 최소 코딩게임 교수전략을 도입한다. 게임 형태의 문제는 학생들에게 흥미와 도전 의식을 불러일으킬 뿐만 아니라 문제를 해결했을 때 성취감도 느낄 수 있다.

나. 외적 학습 동기

외적 학습 동기는 경제적 보상을 통해 학습 동기를 부여받는 것을 말한다. 높은 수준의 내적 학습 동기가 학업성취에 갖는 영향이 큰 것으로 나타나고 있지만, 외적 학습 동기 또한 학업성취에 긍정적인 영향을 미치고 있음에 주목하여 이를 분석한 연구들도 이루어지고 있다(박현정, 2008).

본 연구에서는 수준이 높은 학생들뿐만 아니라 수준이 낮은 학생들에게도 학습 동기를 불러일으키기 위하여 교육과정에 3D VR 게임과 3D 프린트 체험을 제안한다. 교육과정 초반에 [그림 Ⅱ-13]과 같이 최종적으로 완성하고자 하는 경회루를 3D VR 게임을 하는 체험 활동을 경험하도록 함으로써 학생들의 흥미를 유발하고, [그림 Ⅱ-12]와 같이 학생들에게 자유학기제 교육과정이 모두 끝난 후 3D 프린트로 출력물을 제공해줌을 알려줌으로써 경제적 보상을 제공하여 외적 학습 동기를 불러일으킬 수 있다.

Ⅲ. 자유학기제 교육과정 디자인 및 최소 코딩게임 교수전략의 구성

1. 수학과 코딩의 융합 교육과정

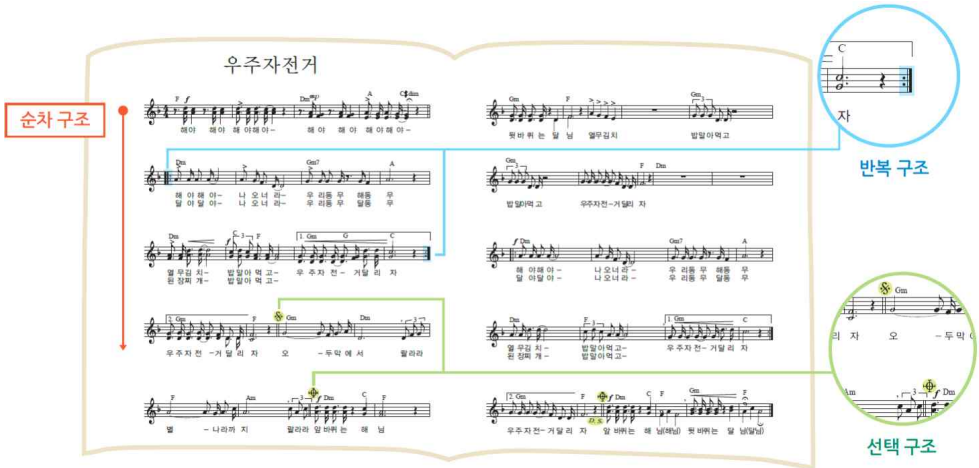
이번 장에서는 앞서 이론적 배경에서 논의한 수학 교과와 정보 교과의 공통 요소를 바탕으로 수학과 정보의 코딩을 융합하는 원리를 분석하고 이를 바탕으로 디자인한 자유학기제 교육과정의 주요 수업 내용 및 학습 과제를 제안한다.

1.1. 수학과 코딩을 융합하는 원리

가. 알고리즘의 제어 구조

정보 교과에는 앞의 <표 Ⅱ-1>에서 제시했듯이 교육과정의 내용 체계에 '알고리즘(Algorithm)'이 포함되어 있다. 알고리즘이란 문제를 해결하는 방법이나 절차를 순서에 따라 논리적으로 설명하거나 표현한 것을 말하며, 이러한 알고리즘을 표현하는 방법으로 순서도가 있다(정영식, 2020). 앞서 언급했듯이 '알고리즘과 순서도'는 제5차 교육과정 수학 교과에 있었던 학습 요소로 수학과 코딩의 공통 요소이다.

코딩에서는 명령어를 처리하기 위해서 여러 가지 조건에 따라 명령어의 순서를 변경하거나 특정 부분을 반복할 필요가 있다. 이러한 역할을 하는 것이 바로 제어 구조이다. 제어 구조에는 순차 구조와 반복 구조, 선택 구조가 있다. [그림 Ⅲ-1]은 정보 교과서에서 설명하고 있는 알고리즘의 제어 구조이다.



[출처: '중학교 음악' 교과서, 세광출판사]

[그림 Ⅲ-1] 정보 교과서에서의 알고리즘의 제어 구조(정영식, 2020)

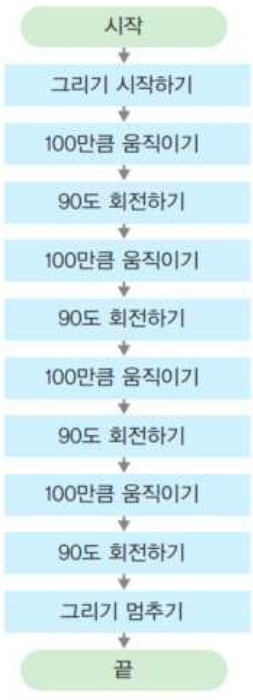
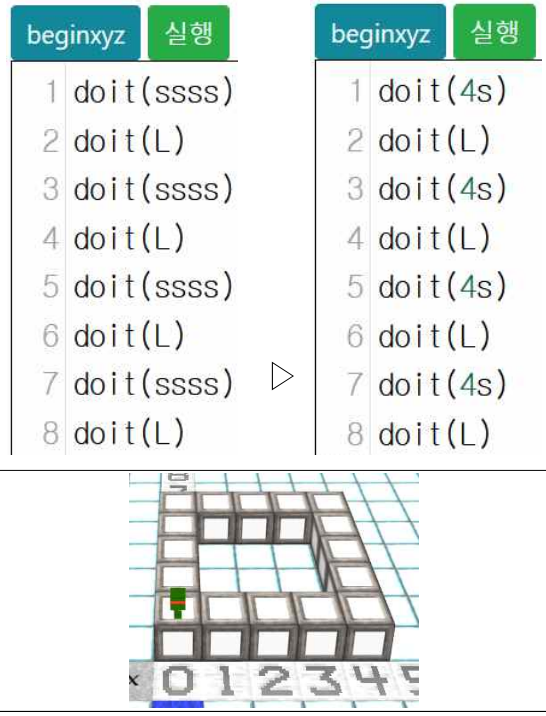
나. 순차 구조와 문자, 계수

알고리즘의 순차 구조는 시작부터 끝까지 제시된 순서에 따라 차례대로 처리하는 경우를 말한다(정영식, 2020).

<표 Ⅲ-1>은 2차원 평면에서 거북이가 100만큼 앞으로 가고 90도 회전하는 행동을 순차적으로 하면서 정사각형을 그리는 과정을 순서도를 이용하여 나타낸 것으로 처음부터 끝까지 순서에 따라 차례대로 처리하는 순차 구조이다. 이를 3차원 좌표공간인 터틀크래프트에서 표현하면 거북이가 'doit(ssss)'를 이용하여 앞으로 4칸 가고 'doit(L)'을 이용하여 왼쪽으로 90도 회전하는 과정을 순차적으로 나타낼 수 있다.

알고리즘의 순차 구조를 순서도로 나타낼 때는 글로 표현하였으나, 터틀크래프트에서는 문자와 숫자를 사용하는 수학적 표현이 사용되었으며, 'ssss'처럼 s를 4번 쓴 것을 '4s'와 같이 수학의 계수를 이용하여 표현할 수도 있다. 여기서 'ssss'가 '4s'로 수학의 계수를 이용하는 과정에서 명령어가 짧아지므로 컴퓨팅 사고력의 추상화 과정이 일어나게 된다.

<표 Ⅲ-1> 순차 구조의 순서도 표현과 터틀크래프트의 순차 구조

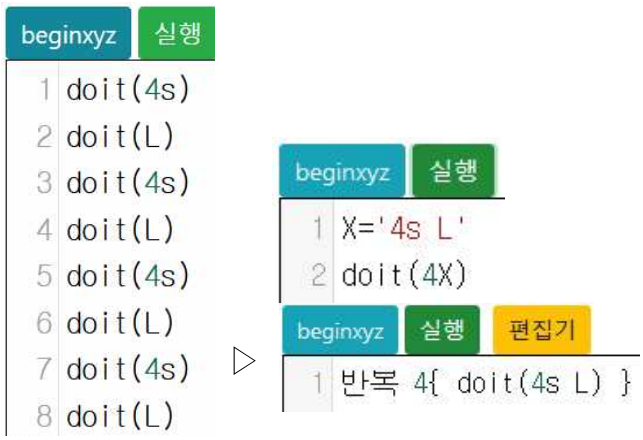
순서도	터틀크래프트
	

다. 반복 구조와 치환 문자

알고리즘의 반복 구조는 주어진 조건을 만족할 때까지 특정 부분을 반복하여 처리하는 경우를 말한다(정영식, 2020). 이러한 반복 구조는 순차 구조를 간략히 줄일 때 사용한다.

<표 Ⅲ-2>는 2차원 평면의 거북이가 100만큼 앞으로 가고 90도 회전하는 것을 4번 반복하여 정사각형을 그리는 과정을 순서도를 이용하여 나타낸 것으로 반복되는 부분을 묶어서 순차 구조를 표현했으므로 반복 구조에 해당한다. 이를 3차원 좌표공간인 터틀크래프트에서 표현하면 거북이가 'doit(4s L)'을 이용하여 앞으로 4칸 가고 왼쪽으로 90도 회전하는 과정을 4번 반복하므로 반복되는 부분인 '4s L'를 X로 묶어서 '4X'로 표현할 수 있다.

<표 Ⅲ-2 > 반복 구조의 순서도 표현과 터틀크래프트의 반복 구조

순서도	터틀크래프트
	

알고리즘의 반복 구조는 4번 반복이라는 단어를 이용하여 표현하였으나, 터틀크래프트에서는 수학의 치환 문자를 사용하여 '4s L'을 X로 치환함으로써 치환 문자의 수학적 표현이 사용되었다. 또한, 터틀크래프트에서는 이러한 반복 구조를 '반복 4 { doit(4s L) }'와 같이 문자와 수식이 들어간 수학적 표현으로도 코딩할 수 있다. '4s'는 s를 5번 쓴 것으로 수학의 계수를 의미하며, '4 { }'는 수학에서의 괄호를 사용하여 괄호 안의 명령어를 4번 반복하는 것을 말한다.

이처럼 순차 구조를 반복 구조로 바꾸는 과정에서 치환 문자를 사용하며 명령어가 훨씬 짧아지게 되는데 마찬가지로 컴퓨팅 사고력의 추상화 과정이 일어나게 된다.

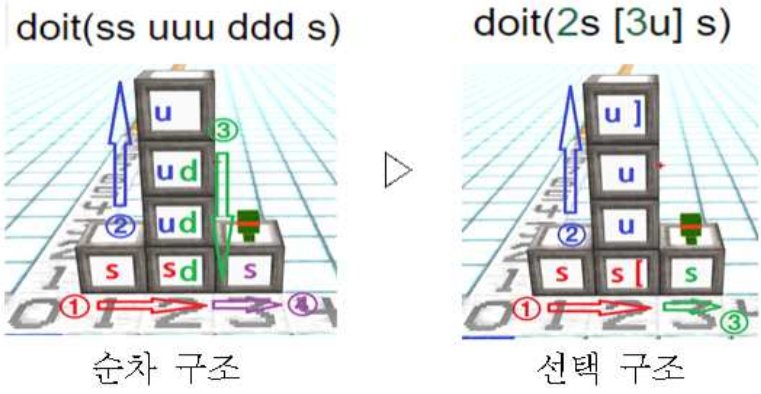
라. 선택 구조와 꺽쇠 명령어

알고리즘의 선택 구조는 주어진 조건에 따라 특정 부분을 선택적으로 처리하는 경우로 정보 교과서에서는 <표 Ⅲ-3>처럼 세뇨와 달세뇨에 비유하여 설명하기도 한다((정영식, 2020).

<표 Ⅲ-3> 알고리즘의 선택 구조

정보 교과서		터틀 크래프트	
에서 세뇨(𝄋)는 반드시 뒤쪽에 달세뇨(D.S)가 나옵니다. D.S를 만나면 𝄋표시가 있는 곳으로 돌아가 거기서부터 연주하라는 뜻입니다.	𝄋	[	거북이의 현재 위치를 기억하는 명령어
	D.S	]	거북이가 기억하는 위치로 돌아가는 명령어

터틀크래프트에서는 세뇨에 해당하는 꺽쇠 명령어 '['와 달세뇨에 해당하는 꺽쇠 명령어 ']'가 있다. '['는 거북이의 현재 위치를 기억하는 명령이고 ']'는 거북이가 기억하는 위치로 돌아가는 명령으로 꺽쇠 명령어를 이용하여 알고리즘의 선택 구조를 표현할 수 있다.



[그림 Ⅲ-2] 터틀크래프트에서 순차 구조와 선택 구조

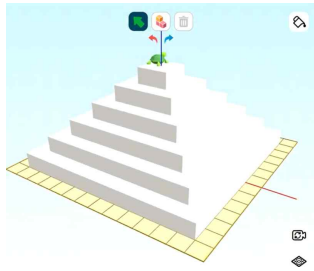
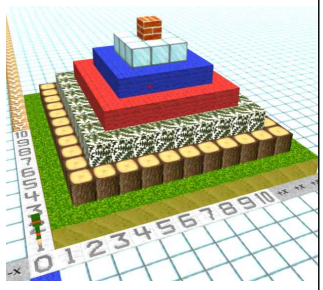
[그림 Ⅲ-2]는 터틀크래프트에서 꺽쇠 명령어를 이용하여 순차 구조를 선택 구조로 표현하는 과정이다. 알고리즘의 순차 구조에서는 ① → ② → ③ → ④의 경로로 거북이가 이동한다. 알고리즘의 선택 구조에서는 거북이가 ①의 경로로 꺽쇠 명령어 '['을 사용하여 현재 위치를 기억하고 ②의 경로로 간 다음에 꺽쇠 명령어 ']'을 사용하여 기억한 위치로 돌아가게 되므로 ③의 경로를 가지 않게 된다. 이처럼 알고리즘의 순차 구조를 선택 구조로 표현하면 거북이의 경로가 짧아져 명령어가 짧아지게 되는데, 이는 컴퓨팅 사고력의 추상화에 해당한다.

마. 집합과 변수

초등학교 6학년 교육과정의 '쌓기나무' 단원에서는 쌓기나무를 쌓아 다양한 입체 구조물을 만드는 과정이 있다. 앞서 언급한 알지오매스와 터틀크래프트는 모두 3차원 좌표공간에서 쌓기나무를 쌓아 입체 구성물을 만들 수 있다. <표 Ⅲ-4>는 알지오매스와 터틀크래프트로 피라미드를 만드는 코드와 그 실행 결과를 나타낸 것이다.

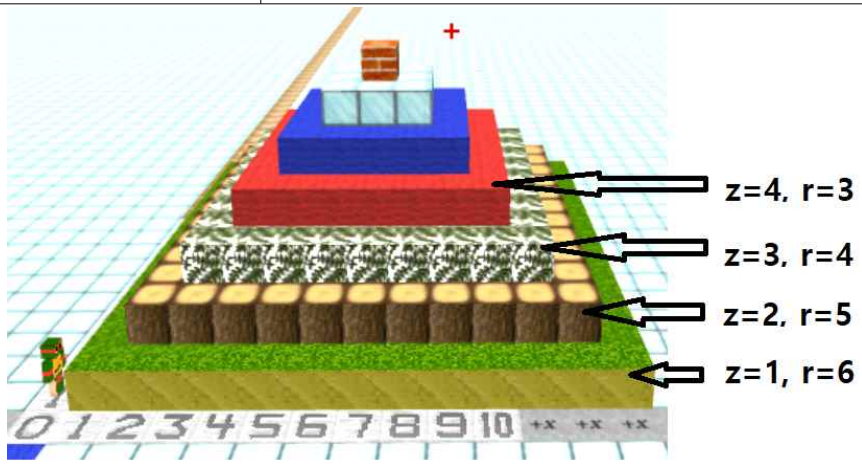
알지오매스를 이용하여 피라미드를 만들기 위해서는 거북이가 직접 쌓기나무를 하나씩 쌓아 올리는 과정을 거쳐야 한다. 쌓기나무를 하나씩 쌓아가며 만드는 방법은 $A=\{2,3,5,7,11,13\}$와 같이 고등학교 교육과정에 등장하는 원소나열법과 같은 수학적 개념에 속한다고 할 수 있다. 터틀크래프트에서도 거북이가 직접 쌓기나무를 하나씩 쌓아 올리는 과정을 거쳐 피라미드를 만들 수도 있지만, 조건에 맞는 쌓기나무를 한 번에 쌓아 올리는 과정으로도 피라미드를 만들 수도 있다. 조건에 맞는 쌓기나무를 한 번에 생성하는 코드는 집합 명령어로서, 이 명령어를 이용하면 코드를 훨씬 짧게 줄일 수 있다. 집합 명령어는 집합 기호 { } 안에 조건을 제시함으로써 해당 조건에 맞는 쌓기나무를 한 번에 쌓으므로 이는 $B=\{x|x$는 15의 양의 약수$\}$ 와 같이 고등학교 교육과정에 등장하는 조건제시법과 같은 수학적 개념에 속한다고 할 수 있다.

<표 Ⅲ-4> 알지오매스와 터틀크래프트의 코드 및 실행 결과

	코드	코드 실행 결과
알지오매스		
터틀크래프트		

집합 기호 { } 안에 조건을 제시함으로써 명령어가 짧아지기 위해서는 컴퓨팅 사고력의 추상화 과정을 거치게 되며, 이때 [그림 Ⅲ-2]와 같이 변수가 사용된다. 구체적으로 $z=1$일 때, $r=6$, $z=2$일 때, $r=5$, $z=3$일 때, $r=4$, … 이므로 $z+r=7$이고 $r=7-z$이다. 이는 두 변수 r, z에 대하여 r은 z에 대한 함수이므로 집합 명령어에서 변수와 함수의 수학적 개념 및 표현이 사용된다.

z = 층	r = 중심으로부터 동서남북으로 확장된 길이
z	$r = 7 - z$



[그림 Ⅲ-3] 집합 명령어에서의 변수

이처럼 터틀크래프트에서는 자바스크립트(JavaScript)를 기반으로 3차원 좌표, 알고리즘, 문자와 계수, 치환, 집합과 논리, 변수와 함수 등 다양한 수학적 개념을 내포한 코드를 통해 자유로운 입체물 구성 활동이 가능하다(정인우, 2020). 또한, 수학 교과의 수학적 표현과 수학적 개념이 포함된 텍스트 코딩으로 정보 교과의 알고리즘 제어 구조의 순차 구조, 반복 구조, 선택 구조를 나타낼 수 있다. 이는 수학 교과와 정보 교과의 코딩을 융합하는 교육과정이며, 교육과정의 목표는 추상화 과정이므로 교육과정을 통해 학생들은 수학 교과의 수학적 사고력과 정보 교과의 컴퓨팅 사고력을 동시에 신장시킬 수 있다.

1.2. 자유학기제 교육과정의 디자인

이번 장에서는 수학 교과와 정보 교과의 코딩을 융합하는 원리를 바탕으로 디자인한 자유학기제 교육과정의 학습 목표 및 학습 내용에 대해 논하겠다. Ⅱ장에서 언급했듯이 일반적으로 자유학기제 교육과정이 프로그램당 일주일에 2~3시간으로 진행되며 총 9차시, 17시간으로 운영되고 있으므로 본 연구에서는 2시간씩 8차시, 총 16시간으로 이루어지는 교육과정을 디자인하도록 한다.

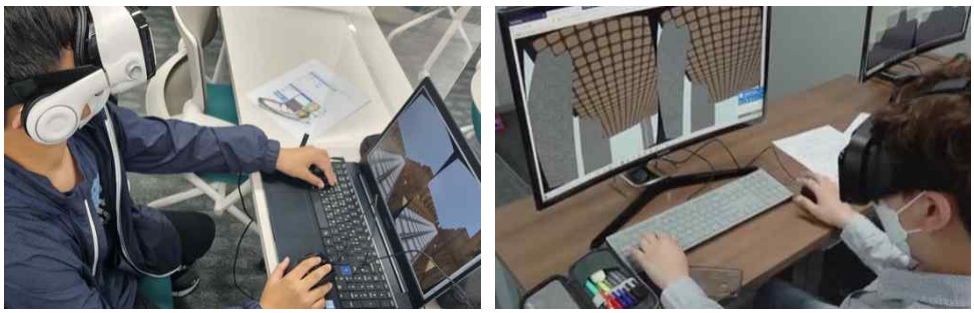
본 연구에서 다루는 자유학기제 교육과정은 '코딩수학' 프로그램이다. '코딩수학' 프로그램은 공통 요소를 바탕으로 수학과 코딩을 융합한 교육과정이며, 본 연구에서는 코딩수학 교육과정이라고 명명한다. 코딩수학 교육과정에서 차시별로 해당하는 수학과 코딩의 학습 요소와 주요 수업 내용은 <표 Ⅲ-5>와 같다.

<표 Ⅲ-5> 코딩수학 교육과정의 주요 수업 내용

단계	차시	주요 수업 내용	수학	코딩
1	1	3차원 좌표공간 및 cube 명령어	좌표, 순서쌍	-
	2	goto 명령어 및 doit 명령어	문자, 계수	알고리즘
	3	회전 명령어	-	순차 구조
2	4	치환 명령어	치환 문자	반복 구조
	5	꺽쇠 명령어	-	선택 구조
3	6	cubeset 명령어 및 집합 명령어	집합, 논리	-
	7	변수 명령어	변수, 함수	변수
	8	최종 과제물 ' 나만의 건축물 만들기'	-	-

단순히 명령어 위주의 연습문제 또는 수학 계산을 하는 알고리즘을 코딩 명령어로 접근하며 수학과 코딩을 물리적으로 결합하는 교육과정은 큰 의미가 없다. 예를 들어, 차시마다 학습 목표가 학습하는 명령어의 숙달로 명령어 위주의 문제를 푼다면 이는 학생의 동기를 유발하기 어렵고 코딩수학 교육과정의 최종 목표인 컴퓨팅 사고력 및 수학적 사고력의 신장을 유도하기도 어렵다. 마찬가지로, 구구단과 같은 수학 알고리즘을 명령어로 프로그래밍하는 단순한 문제 풀이 기반의 교육과정도 학생의 동기유발, 컴퓨팅 사고력의 신장을 끌어내기 어렵다.

이에 본 연구에서 디자인한 코딩수학 교육과정은 학생의 동기유발을 위하여 교육과정 초반에 3D VR 체험을 제공하고 최종적으로 제작한 경회루를 3D 프린트로 제공한다. [그림 Ⅲ-4]는 본 연구의 수업에서 학생들이 3D VR 게임을 체험하는 모습이다.



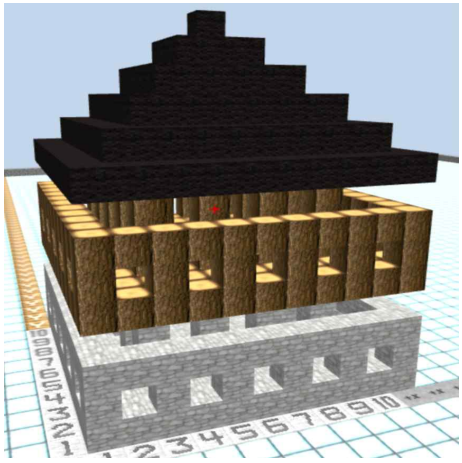
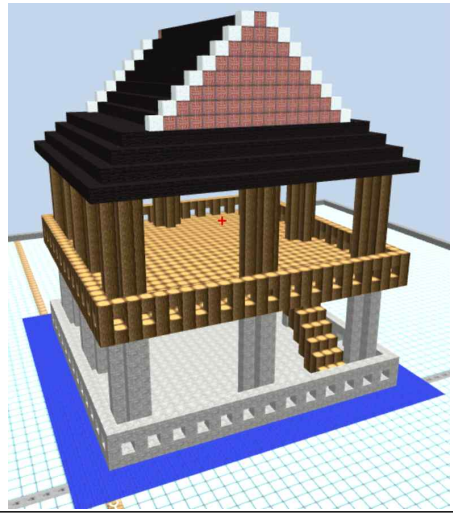
[그림 Ⅲ-4] 3D VR 게임 체험 활동의 실제

또한, 경회루는 기둥, 난간, 바닥, 지붕, 계단 등의 구성요소로 이루어져 있고, 경회루의 구성요소마다 사용하기에 더 효율적이고 적합한 명령어가 존재한다. 매 차시 학생이 흥미와 성취감을 느낄 수 있도록 해당 차시에 배우는 명령어에 적합한 경회루의 구성요소를 제작하여 최종적으로 경회루를 완성하도록 활동 과제를 설계하였다. 이때, 학생의 수준을 고려하여 활동 과제의 난이도 따라 <표 Ⅲ-6>과 같이 기본과 심화, 2가지 형태의 코딩수학 교육과정을 제안한다.

<표 Ⅲ-6> 코딩수학 교육과정의 활동 과제

차시	주요 수업 내용	활동 과제	
		기본	심화
1	3차원 좌표공간 및 cube 명령어	3D VR 게임 체험 경회루 난간 1개 경회루 일자모양 기둥 2개	
2	goto 및 doit 명령어	경회루 난간 5개 경회루 일자모양 기둥 3개	
3	회전 명령어	경회루 난간 1바퀴 경회루 일자모양 기둥 1바퀴	
4	치환 명령어	작은 경회루 난간 및 일자모양 기둥 1바퀴	큰 경회루 난간 및 일자모양 기둥 1바퀴
5	꺽쇠 명령어	일자모양 기둥 1바퀴	십자모양 기둥 1바퀴
		작은 경회루 아래층과 위층	큰 경회루 아래층과 위층
6	cubeset 명령어(기본) 집합 명령어(심화)	경회루 바닥	경회루 바닥 경회루 연못 경회루 지붕하단
7	cubeset 명령어(기본) 집합 명령어(심화) 변수 명령어(심화)	경회루 지붕	경회루 지붕상단 경회루 계단
8	최종 과제물 ‘나만의 건축물 만들기’	최종 과제물 결과 공유 3D 프린트 체험	

앞서 제시한 <표 Ⅲ-5>의 3단계에서는 중학교 1학년 1학기의 수학 교과에서 학습하는 '변수'가 포함되어 있으므로 '변수'라는 수학적 개념의 이해도에 따라 학생의 수준이 차이가 날 수 있다. 이에 1, 2단계에 해당하는 명령어까지 배우는 교육과정을 기본 교육과정으로, 1, 2, 3단계에 해당하는 명령어를 모두 배우는 교육과정을 심화 교육과정으로 구분하여 디자인하였다. 학생들이 배우는 명령어와 경회루 구성요소의 복잡도에 따라 수업 시간에 제공되는 활동 과제의 난이도가 달라진다. 활동 과제는 학생들이 성취감을 느낄 수 있도록 경회루의 구성요소를 제작하여 점차 경회루가 완성되도록 설계하였으며, 기본 교육과정과 심화 교육과정에서 최종적으로 완성되는 경회루의 모양은 [그림 Ⅲ-5]와 같다.

기본 교육과정	심화 교육과정
	

[그림 Ⅲ-5] 기본 교육과정과 심화 교육과정의 경회루

본 연구에서는 수업의 학습 목표가 기본 교육과정에 해당하는 수업 내용으로 기본 교육과정을 적용하였다. 다만, 수준이 높은 학생들을 위해 심화 교육과정의 활동 과제도 학생들이 선택하여 풀 수 있도록 함께 제시하였다. (<부록 1>)

이처럼 경회루 구성요소를 명령어로 만드는 활동은 Papert(1972)의 구성주의(constructionism) 철학을 따르는 것으로 학생들이 조형물을 만드는 과정을 통한 학습(Learning by making)을 실현하는 것이다. 코딩수학 교육과정은 3차원 코딩환경에서 건축물을 만드는 과정을 통해서 자연스럽게 명령에 대한 학습이 이루어지는 Learning by Making 구성주의 교육전략을 기반으로 한다. 이때 말하는 구성주의는 Piaget의 구성주의(constructivism)가 아닌 Papert의 구성주의를 의미한다. Piaget의 구성주의는 개별적 지식의 구성에 초점을 둔다면, Papert의 구성주의는 도구를 통해 지식을 구성하고 학생에게 의미 있는 결과물을 도출하도록 하며, 도구를 매개로 하여 무형의 지식을 유형의 조형물로 만들게 유도한다는 점에서 본 연구의 교육전략과 유사하다.

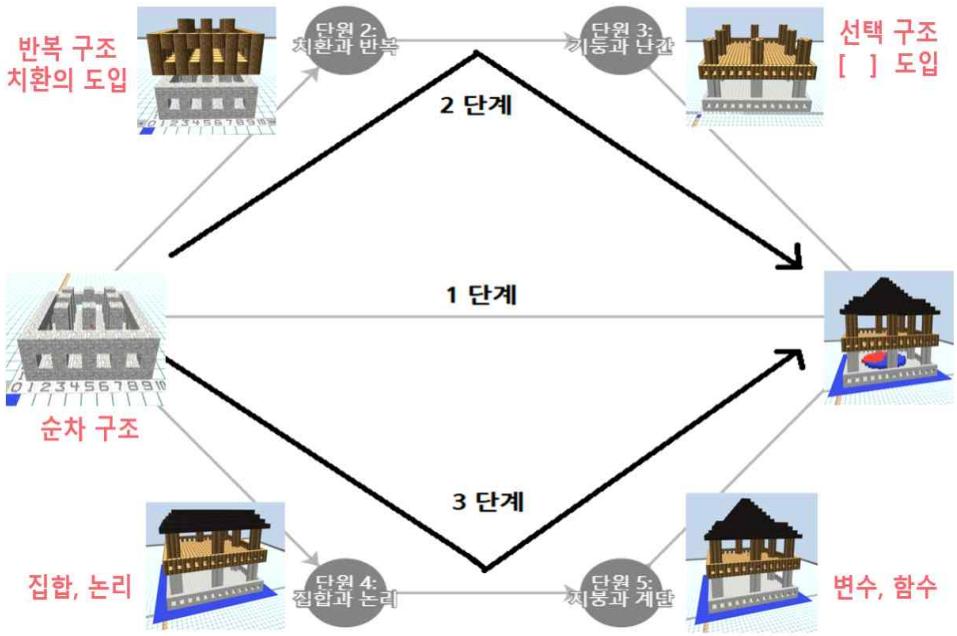
결국 수학과 코딩을 융합한 코딩수학 교육과정의 학습 목표는 최종적으로 학생들의 컴퓨팅 사고력을 신장시켜 수준 상승을 일으키는 것이다. 이를 위해 코딩수학 교육과정은 다음과 같은 2가지 원칙을 바탕으로 교육과정을 디자인하였다.

1) 수준의 차이를 극복하는 교육과정
2) 수준 상승을 지향하는 교육과정

가. 수준의 차이를 극복하는 교육과정

코딩수학 교육과정은 모든 수준의 학생들이 학습 목표를 성취할 수 있는 교육과정으로 디자인한다. 코딩교육은 위계성을 가지고 있는데, 실제로 2015 개정 정보 교과 교육과정의 내용 체계는 네 영역으로 구성되어 있으며 위계성을 가지고 있고, 중학교와 고등학교에도 학교급 간 연계성을 가지고 있다(최정원 외, 2015). 서론에서 언급했듯이, 코딩교육은 수학교육과 관련이 있고, 수학 교과의 중요한 특징은 위계적 성격을 가지므로(신세리, 2017) 코딩교육도 수학교육과 마찬가지로 위계성이 있다. 이렇게 코딩교육이 위계성을 가지면 수학 학습처럼 초반 학습 내용을 제대

로 습득하지 못하면 다음 내용도 이해하기 어렵게 된다. 즉, 코딩교육의 위계성으로 인해 이전 명령어를 이해하지 못하면 다음 수준의 명령어도 이해하기 어렵다. 따라서 학생의 수준에 맞는 명령어를 쓰도록 개별 맞춤형으로 교육과정을 구성해야 한다.



[그림 Ⅲ-6] '코딩수학'의 단계별, 단원별 교육과정

코딩수학 교육과정은 [그림 Ⅲ-6]과 같이 터틀크래프트에서 경회루의 구성요소들 하나씩 만드는 과정을 따라 교육과정을 6단원으로 설계하였다. 다양한 수준의 학생이 모두 경회루를 제작할 수 있도록 수준이 낮은 1단원부터 수준이 가장 높은 6단원까지 수준이 다른 명령어들로 구성되어 있다. 수준이 낮은 학생들도 1단원에서 배우는 명령어를 이용하여 2단원부터 6단원까지의 경회루의 구성요소들을 모두 만들 수 있으며, 학생들은 6단원에 걸쳐 수준이 더 높은 효율적인 명령어를 이용하여 경회루의 구성요소를 하나씩 만들어나가며 최종적으로는 경회루를 완성한다.

나. 수준 상승을 지향하는 교육과정

코딩수학 교육과정은 학생들의 컴퓨팅 사고력을 신장시킬 수 있도록 수준 상승을 지향하는 교육과정으로 디자인한다. 경회루는 구성요소에 따라 사용하기에 더 효율적인 명령어가 있고, 더 효율적인 명령어일수록 명령어의 수준이 더 높으므로 경회루의 구성요소에 따라 수준이 다른 명령어를 도입할 수 있다.

<표 Ⅲ-7>과 같이 기둥과 벽 등 같은 부분이 반복되는 구조물을 만드는 경우는 치환 문자를 이용하여 반복되는 부분을 치환하고 이를 거북이가 직접 하나씩 쌓아나가는 doit 명령어를 사용하는 것이 더 효율적이다. 반면에 탑과 같이 층별로 쌓아지는 구조물을 만드는 경우는 조건에 맞는 쌓기나무를 층별로 한 번에 쌓는 집합 명령어를 사용하는 것이 더 효율적이다.

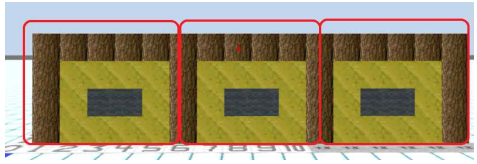
[그림 Ⅲ-6]에서 1단계는 doit 명령어로만 이루어지는 순차 구조를 다루며, 2단계는 치환 문자를 도입하는 반복 구조, 꺽쇠 명령어를 도입하는 선택 구조를 다룬다. 3단계는 집합의 논리와 변수, 함수의 개념을 이용하는 집합 명령어를 다룬다. 따라서 1단계에는 경회루의 기둥을 만들며 가장 수준이 낮은 doit 명령어부터 도입하고 2단계에서 경회루의 바닥, 난간 등으로 점차 더 효율적인 명령어를 도입하며 3단계에는 경회루의 지붕을 만들며 가장 수준이 높은 집합 명령어를 도입한다.

이처럼 차시별로 명령어의 수준이 점차 높아지도록 경회루의 구성요소를 도입하여 학생들의 수준 상승을 유도하는 교육과정으로 설계해야 한다. 학생들은 수준이 낮은 명령어부터 점차 수준이 높은 명령어를 사용하게 되는데, 학생이 수준이 높은 효율적인 명령을 쓴다는 것은 추상화 및 자동화 과정을 거치는 것을 의미하므로 자유학기제 교육과정을 통해 학생의 컴퓨팅 사고력의 신장을 유도함을 알 수 있다.

<표 Ⅲ-7> 경회루의 구성요소에 따른 명령어

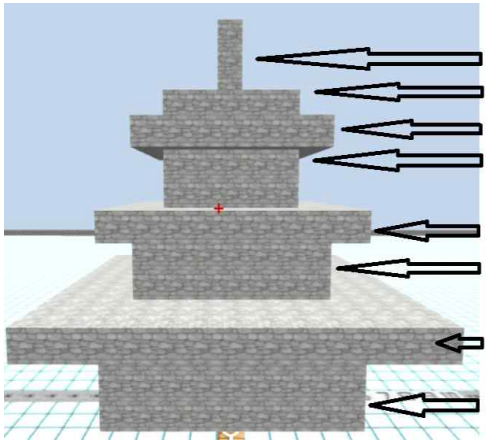
doit 및 치환 명령어	집합 명령어
	

<doit 및 치환 명령어>



```
X='s 3u 5s 3d'
doit(3X ; 2)
doit(4t 2u 3s d ; 11)
doit(2t ; 42)
doit(d 2s ; 11)
```

<집합 명령어>



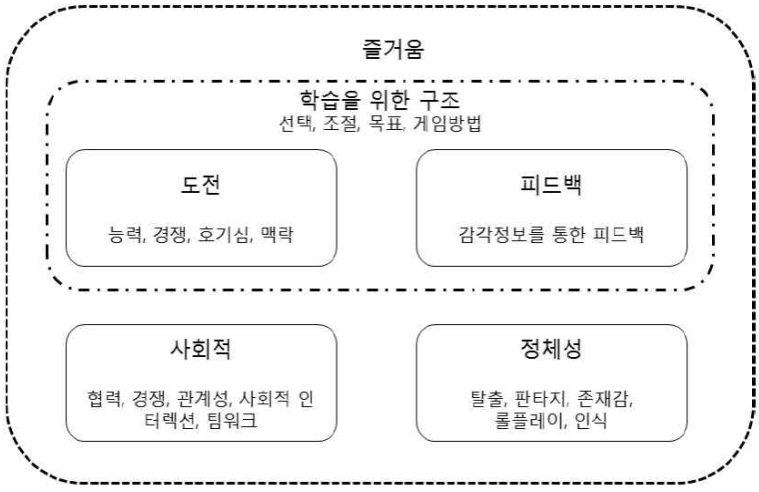
2. 최소 코딩게임 교수전략

2.1. 학습 동기유발 및 수준 상승의 원리

본 연구에서 디자인한 자유학기제 교육과정을 활용하기 위한 최소 코딩게임 교수전략은 다음과 같은 세 가지 원리를 통해 효과적인 지도전략으로 제안한다.

가. 학생의 학습 동기유발 전략

최소 코딩게임 문제는 학생의 동기를 유발하기 위해 게임(game) 형태로 문제를 제시한다. 게임 기반 학습은 학습 목표 달성을 위해 게임이라는 동기유발 전략을 활용하는 형태를 말한다(박남수, 2018).



[그림 Ⅲ-7] 학습자를 위한 게임의 핵심 요소

한국콘텐츠진흥원(2013)은 [그림 Ⅲ-7]과 같이 학습자를 위한 게임의 핵심 요인을 제시하며 도전, 피드백이 있을 때 학습자를 위한 구조가 되며 학습자가 즐거움을 느낄 수 있다고 하였다. 최소 코딩게임은 가장 짧

게 쓰라는 조건을 제시함으로써 학생에게 도전 의식을 느끼게 하며, 학생들은 서로의 응답을 공유하는 과정을 통해 자신의 코드에 대한 피드백을 받을 수 있다. 따라서 최소 코딩게임 교수전략은 학생들에게 도전 의식, 피드백을 제공하므로 이 과정에서 즐거움을 느끼고 학습 동기가 유발될 수 있다. 즉, 게임 형태로 문제를 제시하는 것은 학생이 명령어를 습득할 때 효과적으로 도움을 얻는 방법이라고 할 수 있다.

나. 학생의 수준 상승 유도 전략

최소 코딩게임의 규칙은 주어진 입체 구조물을 만드는 명령어를 가장 짧게 최소(Minimum)로 표현하라는 것이다. 예를 들어, [그림 Ⅱ-7]에서는 스크래치 프로그래밍 언어를 사용하여 정사각형을 만들 때, 블록의 개수를 11개에서 6개로 줄이고 있다. 최소 코딩게임을 활용하면 블록의 개수를 줄이는 것뿐만 아니라 '최소'로 블록의 개수를 사용하자는 조건을 붙여 문제를 제시한다. 반복하는 부분을 주요 아이디어로 정하고 반복 구조를 활용하여 복잡성을 줄이는 과정은 추상화 과정이므로 명령어를 최소로 쓰도록 하는 최소 코딩게임 문제는 학생의 컴퓨팅 사고력을 신장시켜 수준 상승을 유도할 수 있다.

다. 평가 및 피드백 제공 전략

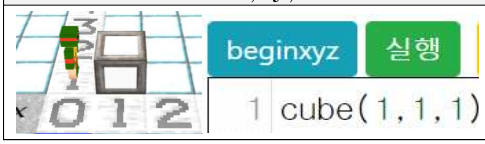
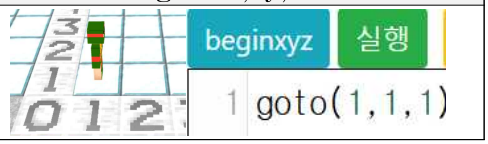
자유학기에서는 평가 결과가 점수로 표기되는 지필평가 위주의 중간고사 또는 기말고사를 실시하지 않는다(교육부, 2015). 평가의 부재는 학생의 동기를 효과적으로 유발하기 어렵다. 이에 최소 코딩게임 전략은 특정 입체 구조물을 만드는 명령어를 '가장 짧게' 코딩하도록 문제를 제시하고 이를 점수화하여 결과를 즉각적으로 알 수 있게 하여 평가의 기능을 제공한다. 이때 학생들은 서로의 응답을 공유하며 스스로 효율적인 코드를 발견하게 되고, 공유된 효율적인 코드를 통해서 자신의 코드를 반성하게 된다(정진환, 2020). 이처럼 최소 코딩게임 전략은 점수화한 결과를 학생들에게 제공하며 동료를 통해 피드백을 받을 수 있다.

2.2. 최소 코딩게임 교수전략을 위한 문제 구성

가. 터틀크래프트에서 최소 코딩게임

본 연구에서는 정진환, 조한혁(2020)이 터틀말에서 활용한 최소코드 게임을 3차원 좌표공간을 추가한 터틀크래프트에서 활용하는 최소 코딩게임을 제안한다. 최소코드 게임은 터틀말에서 적용되는 것으로 3차원 공간이긴 하나 좌표가 도입되지 않는다. 터틀크래프트에서는 좌표공간 및 순서쌍이 추가되므로 이와 관련된 명령어 cube(x, y, z)와 goto(x, y, z)가 등장한다(<표 Ⅲ-8>).

<표 Ⅲ-8> 3차원 좌표공간에서 cube와 goto 명령어

cube(x, y, z)	goto(x, y, z)
	

cube와 goto는 doit과 마찬가지로 하나의 단어이므로 1개로 세며, cube와 goto에는 순서쌍으로 나타낸 좌표가 포함되므로, 그 안의 숫자는 개수로 세지 않는다. 최소 코딩게임의 규칙을 정리하면 다음과 같다.

1. 코드를 가장 짧게 쓸 것
2. 코드의 개수는 글자의 개수로 센다.
3. 숫자 1개, 문자 1개당 글자의 개수 1개로 센다.
4. X=, [,], (,)와 쉼표, 따옴표는 개수로 세지 않는다.
5. X=‘ ’안에 있는 글자는 개수로 세지 않는다.
6. doit, cube, goto 명령어는 1개로 센다.
7. cube와 goto 안에 있는 숫자는 개수로 세지 않는다.

나. 최소 코딩게임 문제의 구성 과정

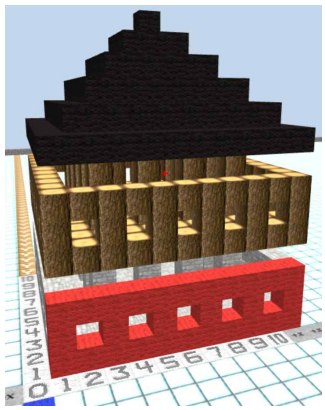
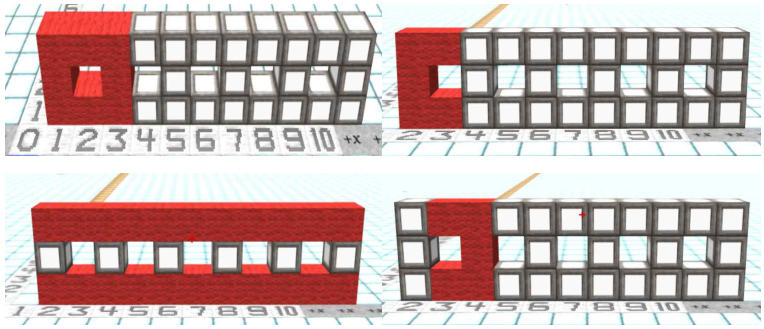
디자인한 자유학기제 교육과정에서 최소 코딩게임 전략을 적용하기 위한 문제는 Ⅱ장의 4.3. 좋은 문제의 요소에 맞게 다음과 같은 원리로 설계되었다.

1) 코딩수학 교육과정에서 최종적으로 만드는 건축물인 경회루의 구성요소를 만드는 문제여야 한다.
2) 학습한 명령어를 활용하면 명령어를 효과적으로 줄일 수 있는 문제여야 한다.
3) 거북이가 다양한 경로로 움직일 수 있는 개방형 문제여야 한다.
4) 추상화 과정을 유도하도록 반복되는 부분이 다양한 방법으로 나타나는 문제여야 한다.
5) 학생들이 즉각적으로 쉽게 해결할 수 없는 도전 의식을 일으키는 문제여야 한다.

위와 같은 설계원리에 맞는 최소 코딩게임 문제의 유형을 구성하기 위하여 2021년도 2학기에 본 연구를 위한 예비 연구로 최소 코딩게임 문제를 설계하여 자유학기제 교육과정에 적용하였다. 첫 번째 설계원리에 의해 최소 코딩게임 문제는 경회루의 구성요소를 만드는 문제여야 한다. 예비 연구에서 적용한 최소 코딩게임 문제의 유형은 <표 Ⅲ-9>와 같이 경회루의 구성요소 중 난간에 해당한다. 또한, 세 번째 설계원리와 네 번째 설계원리에 의해 반복되는 부분이 다양한 방법으로 나타나고 있다.

학생들의 학습 상태는 doit 및 치환, 꺽쇠 명령어까지 배운 상태에서 교수전략을 적용하였다. 최소 코딩게임 예비 문제에 대한 학생들의 예상 코드로서 최소 코딩게임 교수전략 적용 전에는 학생들이 최대 56개의 코드를 쓸 것으로, 교수전략 적용 후에는 최소 12개의 코드를 쓸 것으로 예상하였다.

<표 Ⅲ-9> 최소 코딩게임 예비 문제

<table>
<tr><td>입체
구조물</td><td colspan="2"> 
⇩
  </td><td>경회루 구성요소
난간
학습 상태
doit 및 치환, 꺽쇠 명령어</td></tr>
<tr><td>예상 코드</td><td><교수전략 적용 전>
1 goto(1,1,1)
2 doit(ss uu tt dd ss)
3 doit(ss uu tt dd ss)
4 doit(ss uu tt dd ss)
5 doit(ss uu tt dd ss)
6 doit(ss uu tt dd ss)</td><td><교수전략 적용 후>
1 X='2s [2u 2t 2d]'
2 goto(1,1,1)
3 doit(5X)</td></tr>
<tr><td>예상
코드 개수</td><td>56개</td><td>12개</td></tr>
</table>

최소 코딩게임 예비 문제를 2021년 2학기 자유학기제 교육과정에 적용한 결과, 2가지 문제점이 발생하였다.

첫째, 교수전략을 적용하기 전에 이미 많은 학생이 최소의 개수로 코드를 사용하였다는 점이다. 수준이 높은 학생들뿐만 아니라 중간 정도의 수준인 학생들까지 교수전략을 적용하기 전에 코드의 개수가 12개로 나타났다. 이는 난간을 만드는 문제의 난이도가 학생들의 수준에 비해 낮았기 때문이라고 볼 수 있다. 즉, 앞서 언급한 좋은 문제를 만드는 다섯 번째 설계원리에 해당하지 않는다.

둘째, 수준이 더 높은 명령어를 사용했을 때 코드의 개수와 사용하지 않았을 때 코드의 개수가 똑같다는 점이다. 앞서 <표 Ⅲ-5>에서 언급했듯이, 꺽쇠 명령어는 치환 명령어보다 수준이 더 높은 명령어이다. 그러나 <표 Ⅲ-10>과 같이 꺽쇠 명령어를 사용했을 때와 꺽쇠 명령어를 사용하지 않았을 때 모두 최소코드의 개수가 12개로 같다. 교수전략을 사용하는 목적은 학생의 컴퓨팅 사고력을 신장시켜 더 높은 수준으로의 상승을 유도하기 위함이다. 만약 더 높은 수준의 명령어를 쓰지 않아도 최소로 코드의 개수를 쓸 수 있다면 학생에게 더 높은 수준의 명령어를 사용할 학습 동기를 유발하기 어렵다. 즉, 좋은 문제를 만드는 두 번째 설계원리에 해당하지 않는다.

<표 Ⅲ-10> 최소 코딩게임 예비 문제의 코드 비교

사용한 명령어	doit, 치환 명령어	doit 및 치환, 꺽쇠 명령어
코드	1 X='s 2u 2s 2d t' 2 goto(0,1,1) 3 doit(5X)	1 X='2s [2u 2t 2d]' 2 goto(1,1,1) 3 doit(5X)
최소코드 개수	12개	12개

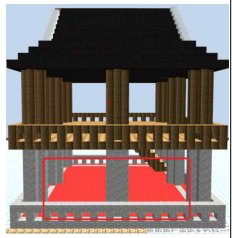
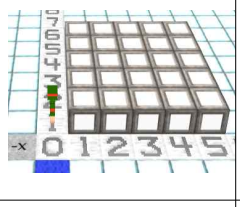
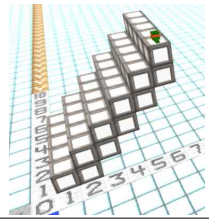
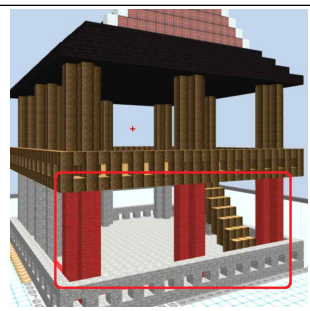
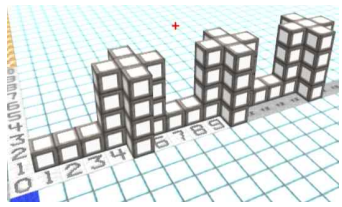
다. 최소 코딩게임 문제의 유형

예비 연구를 실시한 결과, 최소 코딩게임 예비 문제는 문제점이 있었으나 꺽쇠 명령어를 배우지 않고 doit과 치환 명령어까지 배운 학습 상태에서 예제 문제로서 다룬다면 문제점이 발생하지 않으므로 최소 코딩게임 규칙을 설명하기 위한 예제 문제로 사용하는 것을 제안한다.

본 연구에서는 앞서 실시한 예비 연구에서의 2가지 문제점을 개선하여 최소 코딩게임 전략을 적용하기 위한 문제로 총 3가지 유형의 문제를 설계하였다. 3가지 유형의 문제는 각각 경회루의 구성요소인 바닥, 계단, 십자모양 기둥에 해당하는 입체 구조물을 제작하는 문제이다. 이 3가지의 구성요소를 설계한 이유는 다른 구성요소보다 반복되는 부분을 설정할 수 있는 다양한 방법이 존재하여 거북이가 다양한 경로로 움직일 수 있다는 것이다. 또한, 입체 구조물이 단순하지 않고 공간의 여러 방향으로 이동해야 하므로 학생들이 즉각적으로 해결할 수 없는 문제이다. 그뿐만 아니라 수준이 높은 명령어를 사용할수록 명령어를 더 짧게 쓸 수 있어 학생들의 학습 동기를 효과적으로 유발하여 수준 상승을 유도할 수 있다.

본 연구에서는 최소 코딩게임 교수전략의 효과성을 분석하기 위하여 최소 코딩게임 교수전략 적용 전과 후에 문제를 모두 제시하였다. 3차시에는 학생들은 doit 명령어만 학습한 상태로서 최소 코딩게임 교수전략을 적용하기 전, 난이도 하의 경회루 바닥과 난이도 중의 계단을 제작하는 문제를 제시한다. 4차시에는 학생들은 치환 명령어를 학습한 상태로써 최소 코딩게임 교수전략을 적용한 후, 경회루 바닥과 계단을 제작하는 문제를 제시한다. 5차시에는 학생들은 꺽쇠 명령어까지 학습한 상태로서 최소 코딩게임 교수전략 적용 전, 난이도 상의 경회루 십자모양 기둥을 제작하는 문제를 제시하며, 6차시에는 최소 코딩게임 교수전략을 적용한 후, 경회루 십자모양 기둥을 제작하는 문제를 제시한다. 다만, 1기에는 꺽쇠 명령어까지 모두 배운 후 경회루의 바닥, 계단, 십자모양 기둥을 한 번에 제시하였다. 이를 정리하면 <표 Ⅲ-11>과 같다.

<표 Ⅲ-11> 최소 코딩게임 문제 입체 구조물

입체 구조물	 ⇩ 		 ⇩ 		 ⇩ 	
경회루 구성요소	바닥		계단		십자모양 기둥	
난이도	하		중		상	
전략 적용 여부	적용 전	적용 후	적용 전	적용 후	적용 전	적용 후
차시 (1기)	5	6	5	6	5	6
차시 (2기)	5	6	5	6	7	7
학습 상태 (1기)	doit 치환 꺽쇠	doit 치환 꺽쇠 집합	doit 치환 꺽쇠	doit 치환 꺽쇠 집합	doit, 치환, 꺽쇠	doit, 치환, 꺽쇠, 집합
학습 상태 (2기)	doit 치환 꺽쇠	doit 치환 꺽쇠 집합	doit 치환 꺽쇠	doit 치환 꺽쇠 집합	doit, 치환, 꺽쇠, 집합, 변수	doit, 치환, 꺽쇠, 집합, 변수

Ⅳ. 자유학기제 교육과정 및 최소 코딩게임 교수전략의 적용

이번 장에서는 자유학기제 교육과정에서 실시한 '코딩수학' 수업의 연구 대상, 교육과정의 주요 내용 및 과제 설계, 수업 상황, 자료 수집 방법과 분석 방법, 연구 결과 분석에 대해 살펴보고자 한다.

1. 연구 대상

본 연구에서는 서울시 소재 S 중학교 1학년 학생 중 '코딩수학' 수업을 듣는 1기 17명, 2기 18명, 총 35명을 대상으로 하였다. S 중학교에서 실시한 '코딩수학' 수업은 자유학기제 교육과정 중 주제 선택 활동에 해당하며, 2022년 3월 셋째 주부터 5월 둘째 주까지 1기, 5월 셋째 주부터 7월 둘째 주까지 2기 총 2개 반으로 운영되었다. 매주 1차시, 차시 당 2시간씩 총 8주에 걸쳐 16시간의 수업을 진행하였다. 수업은 컴퓨터실에서 컴퓨터로 온라인 코딩환경[3]을 활용하여 진행되었다.

학생들의 특성에 따른 연구 결과를 분석하기 위하여 성별 및 코딩 경험 여부도 함께 파악하였다. 1기는 남학생이 88.2%, 여학생이 11.8%로 남학생이 많았고, 2기는 남학생이 33.3%, 여학생이 66.7%로 여학생이 조금 더 많았다(<표 Ⅳ-1>). 주제 선택 활동은 학생들이 직접 여러 가지 프로그램 중 수업을 듣고 싶은 프로그램을 선택하는데, 먼저 듣고 싶은 수업을 신청하게 되므로 여학생보다 남학생이 '코딩수학' 프로그램에 더 관심을 보였다는 것을 알 수 있다.

3) 온라인 코딩환경 터틀크래프트 홈페이지 URL : snucode.org

<표 Ⅳ-1> 연구 대상의 성별 인원 및 비율

인원(명)	남	여	합계
1기	15	2	17
2기	6	12	18
전체	21	14	35
비율(%)	남	여	합계
1기	88.2	11.8	100
2기	33.3	66.7	100
전체	60	40	100

코딩에 대한 경험 정도는 1기와 2기 모두 비슷한 비율을 보였다(<표 Ⅳ-2>). 특히, 코딩에 대한 경험 정도가 '많이 있다.'와 '조금 있다.'가 88.6%로 대부분 학생이 코딩에 대한 경험이 있었는데, 이는 최근 코딩교육이 강조됨에 따라 많은 학생이 코딩 프로그램을 접해봤다는 점을 알 수 있다.

<표 Ⅳ-2> 연구 대상의 코딩에 대한 경험 정도

인원(명)	많이 있다	조금 있다	없다	합계
1기	4	11	2	17
2기	3	12	3	18
전체	7	24	5	35
비율(%)	많이 있다	조금 있다	없다	합계
1기	23.5	64.7	11.8	100
2기	16.7	66.7	16.7	100
전체	20	68.6	14.3	100

2. 연구 방법 및 절차

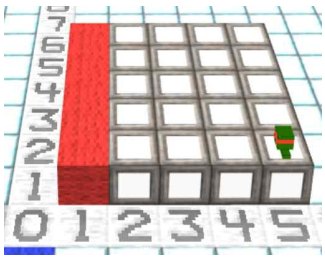
2.1. 최소 코딩게임 문제 설계

가. 최소 코딩게임 문제 (1)

최소 코딩게임 문제 (1)은 난이도 하에 해당하며, 경회루 바닥을 만드는 명령어를 가장 짧은 명령어로 코딩하는 문제이다. 학생들은 최소 코딩게임 형태로 문제를 제시하기 전에는 명령어를 짧게 쓸 필요성을 느끼지 못하므로 명령어를 길게 쓰지만, 최소 코딩게임 형태로 다시 코딩하도록 하였을 때, 도전 의식을 느끼고 명령어를 최소로 쓸 수 있다.

<표 Ⅳ-3>은 최소 코딩게임 문제 (1)에 대한 학생들의 예상 코드로서 최소 코딩게임 교수전략 적용 전에는 학생들이 최대 50개의 코드를 쓸 것으로, 교수전략 적용 후에는 최소 7개의 코드를 쓸 것으로 예상하였다. 터틀크래프트를 처음 실행했을 때, 거북이의 위치는 (0, 1, 1)이므로 학생들이 goto(0, 1, 1)를 맨 처음에 쓰지 않더라도 쓴 것으로 간주하였다.

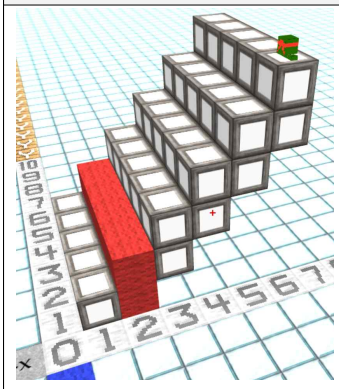
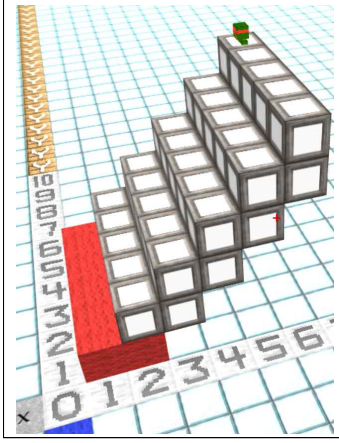
<표 Ⅳ-3> 최소 코딩게임 문제 (1)

최소 코딩게임 교수전략 적용 여부		예상 코드	예상 코드 개수
	적용 전	goto(0,1,1) doit(sssss tttt l) doit(ssss tttt l) doit(ssss tttt l) doit(ssss tttt l) doit(ssss tttt)	50개
	적용 후	goto(0,1,1) X='s [4l]' doit(5X)	7개

나. 최소 코딩게임 문제 (2)

최소 코딩게임 문제 (2)는 난이도 중에 해당하며, 경회루 계단을 만드는 명령어를 가장 짧은 명령어로 코딩하는 문제이다. <표 Ⅳ-4>는 최소 코딩게임 문제 (2)에 대한 학생들의 예상 코드로서 최소 코딩게임 교수전략 적용 전에는 학생들이 최대 105개의 코드를 쓸 것으로, 교수전략 적용 후에는 최소 13개의 코드를 쓸 것으로 예상하였다.

<표 Ⅳ-4> 최소 코딩게임 문제 (2)

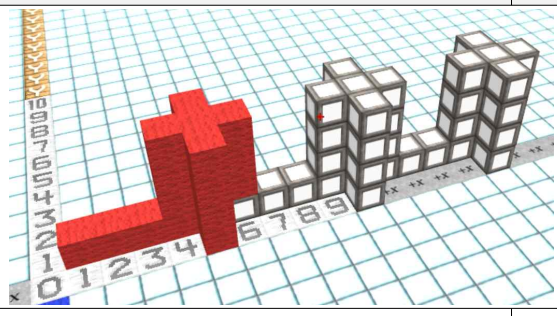
최소 코딩게임 교수전략 적용 여부		예상 코드	예상 코드 개수
	적용 전	goto(0,1,1) doit(s llll rrrr) doit(s llll rrrr u llll rrrr) doit(s llll rrrr u llll rrrr) doit(s llll rrrr u llll rrrr) doit(s llll rrrr u llll rrrr) doit(s llll rrrr u llll rrrr)	105개
	적용 후	goto(0,1,1) X='s [4l] u [4l]' doit(s [4l] 5X) goto(0,1,1) X='[4l] s [4l] u' doit(s 5X 4l)	13개

다. 최소 코딩게임 문제 (3)

최소 코딩게임 문제 (3)는 난이도 상에 해당하며, 경회루 십자모양 기둥을 만드는 명령어를 가장 짧은 명령어로 코딩하는 문제이다. <표 Ⅳ-5>는 최소 코딩게임 문제 (3)에 대한 학생들의 예상 코드로서 최소 코딩게임 교수전략 적용 전에는 학생들이 최대 135개의 코드를 쓸 것으로, 교수전략 적용 후에는 최소 19개의 코드를 쓸 것으로 예상하였다.

최소 코딩게임 문제 (3)은 반복되는 부분의 설정보다 반복되는 부분 안에서 꺽쇠 명령어를 어떻게 설정하느냐에 따라 명령어의 길이가 달라진다. 반복되는 부분의 설정은 반복 구조이고, 꺽쇠 명령어의 사용은 선택 구조이므로 수준 상승을 유도하는 문제임을 알 수 있다.

<표 Ⅳ-5> 최소 코딩게임 문제 (3)

<table>
<tr><th rowspan="2">최소
코딩게임
교수전략
적용 여부</th><th>예상 코드</th><th>예상
코드 개수</th></tr>
<tr><td>  </td><td></td></tr>
<tr><td>적용 전</td><td>

```
goto(0,1,1)
doit(sssss uuu ddd t uuu ddd s)
doit(r uuu ddd l l uuu ddd r s uuu ddd t)
doit(sssss uuu ddd t uuu ddd s)
doit(r uuu ddd l l uuu ddd r s uuu ddd t)
doit(sssss uuu ddd t uuu ddd s)
doit(r uuu ddd l l uuu ddd r s uuu ddd t)
```

</td><td>135개</td></tr>
<tr><td>적용 후</td><td>

```
goto(0,1,1)
X='5s [3u] [t 3u] [r 3u] [l 3u] [s 3u]'
doit(3X)
```

</td><td>19개</td></tr>
</table>

2.2. 자유학기제 수업 상황

가. 수업 방법

본 연구에서 진행하는 수업 방식은 코로나19로 인한 오프라인과 온라인 수업을 병행한다는 가정하에 오프라인 수업과 온라인 수업이 모두 가능하도록 구상하였다. 이를 위해 [그림 Ⅳ-1]과 같이 터틀크래프트를 활용하는 온라인 홈페이지[4]에서 과제 게시판을 이용하여 학생들이 과제를 제출할 수 있도록 하였다. 오프라인 수업은 컴퓨터실에서 학생 1명당 1대의 컴퓨터로 수업을 진행하였으며, 온라인 수업은 실시간 쌍방향 수업으로 진행하였다.

[그림 Ⅳ-1] 터틀크래프트 온라인 홈페이지 과제 게시판

4) 터틀크래프트 홈페이지 URL : codingmath.org

나. 수업 형태

최소 코딩게임 교수전략의 목적은 학생들의 학습 동기 및 경쟁심, 도전 의식을 유발하며 결과를 점수화하여 동료 학생들과 결과를 공유하여 즉각적으로 피드백을 받아 수준 상승을 유도하기 위함이다. 최소 코딩게임 문제를 제시했을 때, 수준이 낮은 학생들은 푸는 것을 어려워하는 경향이 있으며, 모둠으로 활동할 때 수준이 높은 학생이 수준이 낮은 학생을 코드를 설명해줄 수 있다는 점에서 최소 코딩게임 문제를 2인 1조 모둠별로 풀도록 수업을 진행하였다.

다. 문제 제시 형태

1기에는 명령어를 모두 배운 상태에서 최소 코딩게임 전과 후를 비교하기 위해 5차시까지 doit, 치환, 꺽쇠 명령어를 모두 배운 후 최소 코딩게임 문제를 제시하고, 이후 6차시에서 교수전략을 적용한 후 최소 코딩게임 문제를 제시하였다(<표 Ⅳ-6>).

1기에서 한 차시에 최소 코딩게임 문제 3문항을 모두 해결하기에 시간상으로 부족함이 있어 2기에는 문제 (1), (2)과 문제 (3)을 나누어 제시하였다. 5, 6차시에는 문제 (1), (2)을, 7차시에는 문제 (3)을 최소 코딩게임 교수전략 적용 전과 후로 나누어 제시하였다.

<표 Ⅳ-6> 최소 코딩게임 문제 제시 일정 비교

차시	최소 코딩게임 문제	
	1기	2기
5	최소 코딩게임 문제 (1), (2), (3) - 교수전략 적용 전	최소 코딩게임 문제 (1), (2) - 교수전략 적용 전
6	최소 코딩게임 문제 (1), (2), (3) - 교수전략 적용 후	최소 코딩게임 문제 (1), (2) - 교수전략 적용 후
7		최소 코딩게임 문제 (3) - 교수전략 적용 전, 후

2.3. 자료 수집 및 분석

분석 대상이 되는 학생들의 자료는 3차원 좌표계 코딩환경인 터틀크래프트 홈페이지를 이용하여 수집한 학생들의 명령어, 학생들이 명령어를 직접 수기로 작성한 활동지, 구글 설문조사로 실시한 사전 설문지와 사후 설문지, 수업 시간 직후 실시한 사후 인터뷰로 이루어져 있다.

가. 최소 코딩게임 교수학습 전략에 따른 학생들의 명령어 변화과정

최소 코딩게임 교수학습 전략은 동일한 입체 구조물을 만드는 명령어를 학생들이 최대한 짧게 쓰도록 유도하기 위하여 명령어의 개수를 세는 게임 활동 형식으로 지도하는 전략을 말한다(정진환, 2015). 코딩 명령어를 학습하며 향상되는 컴퓨팅 사고력은 추상화, 자동화를 핵심 개념으로 하고 있는데, 추상화 및 자동화는 명령어를 효율적으로 짧게 쓰는 것을 의미한다. 따라서 명령어를 효율적으로 짧게 쓰도록 유도할 때, 학생들의 추상화 및 자동화 과정이 발생하므로 컴퓨팅 사고력을 신장시키는 수준 상승이 일어난다.

먼저 최소 코딩게임 교수전략을 사용하지 않고 학생들이 입체 구조물을 만드는 명령어를 코딩하도록 지도한다. 그다음 최소 코딩게임 교수전략을 사용하여 동일한 입체 구조물을 만드는 명령어를 코딩하도록 지도한다. 학생들은 자신이 코딩한 명령어를 활동지에 직접 적고 코드의 개수를 세어보며, 코드를 더 짧게 만들기 위한 활동을 한다. 이때 학생들이 활동지에 작성한 명령어를 바탕으로 최소 코딩게임 교수전략을 사용하기 전과 사용한 후 명령어의 개수를 비교한다. 또한, 명령어의 개수가 짧은 학생들과 명령어의 개수가 긴 학생들의 명령어를 비교하여 명령어를 짧게 쓰기 위해 학생들이 사용한 핵심적인 사고 과정을 분석한다. 이를 위해 터틀크래프트 홈페이지에서 학생들이 명령어를 실행할 때마다 자동으로 명령어가 저장되는 기능을 활용한다.

나. 수학과 코딩의 인식에 관한 사전·사후 설문조사

사전 설문조사는 자유학기제 주제 선택 수업 첫 시간에 실시하며, 사후 설문조사는 마지막 수업 때 실시한다. 설문조사는 5점 척도를 기준으로 하는 평정척도법이며, 총 13개의 문항으로 구성되어 있다. 수학과 코딩에 대한 인식 변화와 태도 조사는 '컴퓨팅 사고력에 대한 인식 변화와 과학 학습에 대한 태도 조사'(황요한, 2016) 선행연구를 바탕으로 제작되었다. 크게는 '1) 코딩에 대한 자신감, 2) 수학과 코딩에 대한 유용성, 3) 수학과 코딩의 관련성 4) 문제에 대한 자신감 및 과제집착력'의 4가지 영역으로 구성되어 있다.

문항별로 평균을 구한 후 사전 설문조사와 사후 설문조사 간의 변화를 살펴본다. 사전 설문조사에서 성별도 함께 조사하여 성별에 따라 문항별로 평균을 각각 구한 후 유의미한 차이가 있는지 분석한다. 또한, 2기 사후 설문조사에서 수학과 코딩의 관련성에 관한 문항을 서술식 1문항을 추가로 조사하여 구체적인 인식 및 생각을 분석한다.

다. 수업 종료 후 진로 탐색의 변화

코딩수학 수업이 모두 종료된 후 학생들의 진로 탐색에 관한 변화를 살펴보기 위하여 사후 설문조사를 실시한다. 코딩수학 교육과정에 참여한 전체 학생들을 대상으로 실시하며, 주요 내용은 고등학교 선택 과목인 '인공지능 수학'의 관심도이다. 5점 척도 객관식 1문항과 서술식 1문항을 직접 글로 적도록 하였으며, 최소 코딩게임 문제에 대한 응답을 분석하여 수준을 파악한 뒤, 수준이 높은 학생과 수준이 낮은 학생에 대하여 추가적인 심층 인터뷰를 실시하였다.

3. 연구 결과

3.1. 최소 코딩게임의 결과

이번 장에서는 최소 코딩게임의 결과로 먼저 전체 문제에 대하여 코드의 개수 변화 및 치환 문자의 사용 변화를 살펴본 후, 각 문제에 대하여 코드를 자세히 분석한다. 이때 명령어를 짧게 줄이기 위한 핵심 아이디어를 바탕으로 코드를 유형화하여 학생들의 코드를 분석하고 이를 논하고자 한다.

가. 최소 코딩게임 교수전략에 따른 결과

최소 코딩게임 교수전략은 학생들이 주어진 입체 구조물을 만드는 명령어를 가장 짧게 코딩하는 교수전략이다. 명령어를 짧게 쓰기 위해서는 반복되는 부분을 발견하고 이를 치환 문자를 이용하여 명령어를 줄이는 과정이 필연적으로 일어나야 한다.

문제 (1), (2), (3)에 대하여 코드의 개수 및 치환 문자의 사용 변화를 정리하면 <표 Ⅳ-7>과 같다.

<표 Ⅳ-7> 코드의 개수 및 치환 문자의 사용 변화

		코드의 개수			치환 문자 사용	
		감소	변화 없음	증가	적용 전	적용 후
문제 (1)	인원(명)	11	6	0	14	17
	비율(%)	58.8	41.2	0	82.4	100
문제 (2)	인원(명)	8	9	0	14	17
	비율(%)	47.1	52.9	0	82.4	100
문제 (3)	인원(명)	13	3	1	16	17
	비율(%)	76.5	17.6	5.9	94.1	100

구체적으로 문제 (1), (2), (3)에 대하여 2인 1조 모둠별로 코드의 개수 변화량을 살펴보면 <표 Ⅳ-8>과 같다. 분석의 편의를 위해 1기 학생들 2인 1조는 g1, g2, …, g8과 같이 지칭하고, 2기 학생들 2인 1조는 g9, g10, …, g17과 같이 지칭한다.

<표 Ⅳ-8> 학생 모둠별 코드의 개수 변화량

모둠	문제 (1)			문제 (2)			문제 (3)		
g1	20	15	-5	44	19	-35	59	23	-36
g2	7	7	0	13	13	0	22	23	+1
g3	7	7	0	8	8	0	30	19	-11
g4	7	7	0	14	14	0	46	19	-27
g5	20	13	-7	12	12	0	34	29	-5
g6	24	18	-6	26	26	0	32	23	-9
g7	14	10	-4	13	13	0	24	23	-1
g8	7	7	0	44	14	-30	30	30	0
g9	7	7	0	13	13	0	25	21	-4
g10	16	16	0	27	16	-11	34	24	-10
g11	14	10	-4	23	13	-10	24	19	-5
g12	17	10	-7	13	13	0	50	29	-21
g13	12	10	-2	16	13	-3	21	20	-1
g14	23	16	-7	25	13	-22	29	23	-6
g15	17	11	-6	29	14	-15	35	35	0
g16	13	10	-3	19	13	-6	25	23	-2
g17	20	13	-7	13	13	0	25	25	0

나. 최소 코딩게임 문제 (1) 분석 결과

최소 코딩게임 문제 (1)은 경회루 바닥을 만드는 명령어를 코딩하는 문제이다. <표 Ⅳ-9>는 최소 코딩게임 문제 (1)에 대하여 교수전략을 적용하기 전과 적용한 후의 학생들의 답안이다.

<표 Ⅳ-9> 최소 코딩게임 문제 (1) 학생 답안

모듬	최소 코딩게임 적용 전		최소 코딩게임 적용 후	
	코드	코드 개수 (유형)	코드	코드 개수 (유형)
g1	1. goto(0,1,1) 2. dott(5s) 3. goto(0,2,1) 4. dott(5s) 5. dato(0,3,1) 6. dott(5s) 7. goto(0,4,1) 8. dott(5s) 9. goto(0,5,1) 10. dott(5s)	20 (⑤)	1.goto(1,1,1) 2. X='4s L' 3. dott(4X) 4.goto(2,2,1) 5. X='2s L' 6. dott(4r) 7. cube(3,3,1)	15 (③)
g2	goto(1,0,1) X='l[4s]' doit(5X)	7 (②)	goto (1,0,1) X = 'l[4s]' doit(5X)	7 (②)
g3	X='s[4l] dott(5X)	7 (①)	X ='s[4l]' dott(5X)	7 (①)
g4	goto(1,1,1) X='s[4l] doit (5X)	7 (①)	goto(0_1,1) X='s[*l]' doit(5x)	7 (①)
g5	goto(0,1,1) doit(5s 4l 4t 3r 3s 2l 2t 1r 1s)	20 (③)	goto(0,1,1) X='[4l 4r t]' doit(5s 4X 4l)	13 (①)

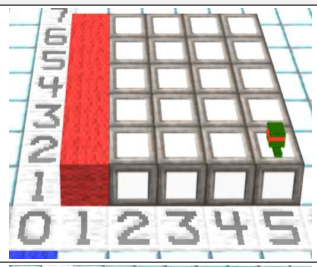
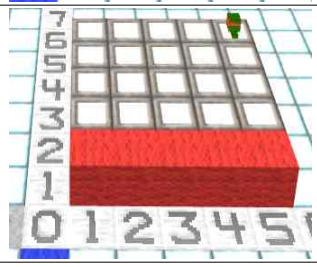
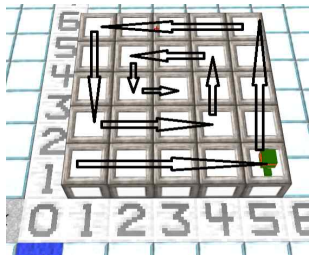
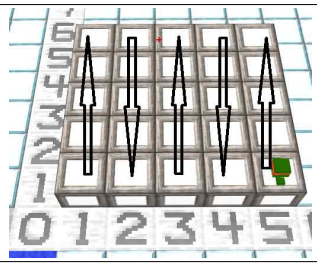
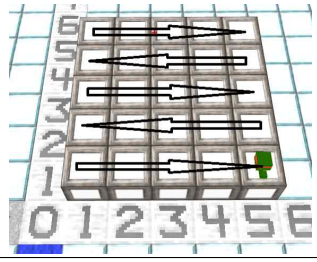
g6	goto(0,1,1) X='4s L' doit(4X) goto(1,2,1) doit(2s L 2s L 2s L 2s L) goto(1,3,1) doit(5)	24 (③)	goto(0,1,1) goto(0,3, X='5s L' doit(X) doit(X) goto(2,4,1 goto(0,2,1) doit(X) doit(X) goto(0,5,	18 (⑤)
g7	X='[5s]' doit(5X l X l X l X l X)	14 (②)	X='[45] l' doit(s 4X 4 5)	10 (②)
g8	goto(0,1,1) X='s[4l]' doit(5X)	7 (①)	goto(0,1,1) X='s[4l]' doit(5X)	7 (①)
g9	goto(1,0,1) X='l[45]' doit(5X)	7 (②)	X='s[4l]' doit(5X)	7 (①)
g10	goto(0,1,1) X='L s L 4s R s R 4s' doit(5s 2X)	16 (⑤)	goto(0,1,1) X='L s L 4s R s R 4s' doit(5s 2X)	16 (⑤)
g11	X='[45]' doit(5) doit(X l X l X l X l X)	14 (②)	X='[45] l' doit(5 4X 4 5)	10 (②)
g12	goto(0,1,1) X='5s' doit(X) goto(0,2,1) doit(X) (이하 생략)	17 (⑤)	goto(0,1,1) X='5L[45]R' doit(5X)	10 (④)
g13	X='[45] l l' doit(1s 4X [45])	12 (②)	X='[45] l' doit(5 4X [45])	10 (②)
g14	goto(0,1,1) X='4s L' doit(5 3X 3s L 3s L 2s L 2s L sL s)	23 (③)	goto(1,0,1) X='4s L' Y='2s L' doit(4X s l 4Y s l)	16 (③)

g15	goto (1,1,1) X='45 4t 1' doit (4X) goto (2,2,1) X='25 4t 1' doit (4Y) goto (2,3,1) doit (5)	17 (③)	goto (1,1,1) X='45 4t 1' doit (4X 45)	11 (②)
g16	goto(1,1,1) X='45 4t 1' doit = (4X, 45 4t)	13 (②)	goto (0,1,1) X='45 4t 1' doit (5 4X 45)	10 (②)
g17	goto (0,1,1) doit (55) goto (0,2,1) doit (55) goto(0,3,1) doit (55) goto (0,4,1) doit (55) goto (0,5,1) doit (55)	20 (②)	goto (0,1,1) X='545,4t' doit (2X54)	13 (①)

학생들의 답안을 반복되는 부분을 어떻게 인지하느냐에 따라 총 5가지 형태로 유형화하였다. 유형 ①은 반복되는 부분을 x축과 평행하게 인지하여 치환한 유형이고, 유형 ②는 반복되는 부분을 y축과 평행하게 인지하여 치환한 유형이며, 유형 ③은 반복되는 부분을 인지하지 못하고 거북이가 정사각형의 바깥부터 안으로 회전하며 이동하는 경로를 표현한 유형이다. 유형 ④는 거북이가 x축과 평행하게 이동하지만, 같은 방향으로 이동하지 않고 좌우로 이동하는 경로를 표현한 유형이며, 유형 ⑤는 거북이가 y축과 평행하게 이동하지만, 같은 방향으로 이동하지 않고 앞뒤로 이동하는 경로를 표현한 유형이다. 여기서 거북이가 x축과 평행하게 이동하지만, 치환을 사용하지 않고 goto 명령어를 사용하여 이동한 경우는 반복되는 부분을 인지하지 못하였으므로 유형 ⑤로 보았다.

즉, 유형 ①과 ②는 반복되는 부분을 인지한 경우이고, 유형 ③, ④, ⑤는 반복되는 부분을 인지하지 못한 경우이다. <표 Ⅳ-10>을 보면, 반복되는 부분을 인지한 경우는 최소코드의 개수가 7개로 가장 짧다. 반면에 반복되는 부분을 인지하지 못한 유형 ③, ④, ⑤의 경우는 최소코드의 개수가 각각 15개, 10개, 16개로 짧지 않다. 이는 주요 아이디어를 파악하여 복잡성을 줄이는 추상화 과정을 실행하지 못했기 때문이다.

<표 Ⅳ-10> 최소 코딩게임 문제 (1) 분석 결과

유형	거북이의 경로	반복 인지 여부	최소코드 (적용 후 평균)	적용 전 학생 수 (비율)	적용 후 학생 수 (비율)
①		O	7 (9)	3 (17.6)	6 (35.3)
②		O	7 (9.6)	7 (41.2)	6 (35.3)
③		X	15 (15.5)	4 (23.5)	2 (11.7)
④		X	10 (10)	0 (0)	1 (5.9)
⑤		X	16 (17)	3 (17.6)	2 (11.8)

최소 코딩게임 교수전략을 적용한 후, 코드의 개수가 감소한 비율이 58.8%, 변화 없는 비율이 41.2%, 증가한 비율이 0%였으며, 이중 유형 ①과 ②으로 해결한 학생의 비율이 70.6%로 많은 학생이 반복되는 부분을 인지한 것으로 나타났다. 이는 최소 코딩게임 교수전략이 학생에게 추상화 과정을 사고하도록 유도하여 수준 상승을 끌어내는 효과적인 교수전략임을 알 수 있다.

최소 코딩게임 적용 후 학생들의 핵심 사고 과정을 심층적으로 살펴보기 위해 코드의 개수가 가장 많이 변화하면서도 유형 ③에서 반복되는 부분을 인지하는 유형 ①로 변화한 2인 1조 학생 g5에게 수업 시간 직후 심층 질문을 하였다.

연구자 : 문제 (1)에서 코드의 개수를 줄이려고 어떻게 한 거야?
g5 : 치환을 사용했어요. 아, 선생님이 말씀해주셔서 1도 없앴어요.
연구자 : 최소 코딩게임 전에는 치환을 왜 사용하지 않았니?
g5 : 그냥…. 거북이가 뱅글뱅글 움직이면서 가는 게 더 쉬웠어요.
연구자 : 최소 코딩게임 때는 왜 치환을 사용하였니?
g5 : 짧게 쓰려면 치환을 사용해야 짧게 쓸 수 있어서요.
연구자 : 그렇다면 어떤 부분을 치환한 거니?
g5 : 세로로 한 줄씩 왼쪽으로 갔다가 오른쪽으로 가는 과정이요.
연구자 : 꺽쇠 명령어를 이용해서 '[4l 4r t]'로 썼는데, 꺽쇠 명령어 '['의 역할이 무엇인지 기억나니?

… (중략) …

g5 : 아! '4r'는 지워도 돼요.
연구자 : 그래, '[4l] t'로 잘 고쳤네. 그렇다면 지금 명령어에서 더 짧게 만들 수는 없을까?
g5 : …. 잘 모르겠어요.
연구자 : 음, 명령어가 doit(5s 4X 4l)인 것을 보니 거북이가 앞으로 5칸 먼저 갔다가 뒤로 1칸씩 움직이면서 세로 한 줄씩 만드는구나.

연구자 : 앞으로 5칸 갔다가 뒤로 1칸씩 가게 되면 거북이가 쌓기나무를 쌓은 곳에 다시 쌓게 되네. 앞으로 5칸을 먼저 가야 할까?

… (중략) …

g5 : 선생님, '[4l] t'를 '[4l] s'로 고치고, doit(5s 4X 4l)을 doit(4X 4l)로 고쳤어요. 그런데 모양이 이상하게 나와요.

연구자 : 맨 앞에 1칸이 비어 있네. 그렇다면 왼쪽으로 4칸 먼저 가고 앞으로 1칸 가는 것이 아니라 그 반대가 되어야 하지 않을까?

g5 : 그러면 '[4l] s'가 's [4l]'이 되는 거예요?

연구자 : 그렇지! 명령어를 직접 실행해보고 doit 안에도 바꿔보렴.

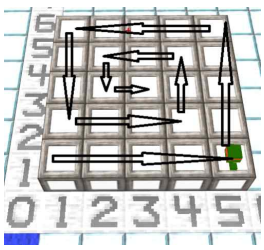
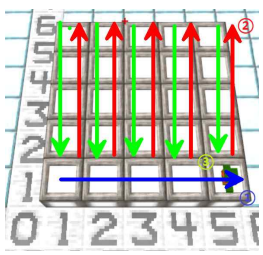
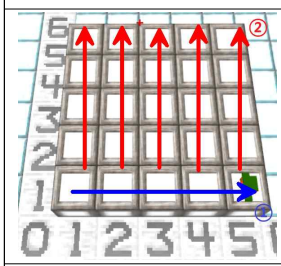
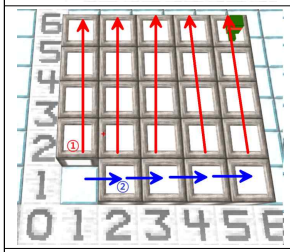
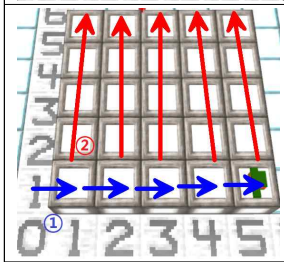
… (중략) …

연구자 : 잘했어. 만약 더 큰 바닥을 만들려면 어디를 수정하면 될까?

학생 g5는 최소 코딩게임 적용 전에는 반복되는 부분을 인지하지 못하고 단순하게 거북이의 이동 경로를 따라 코드를 썼지만, 명령어를 짧게 쓰기 위해 거북이의 경로를 반복되는 부분을 인지하여 치환을 사용하는 모습을 보였다. 치환을 사용하지 않으면 경회루의 바닥을 더 크게 만들 때 계수만 바꿔서 명령어를 쓰는 것이 불가능하다. 치환을 사용하면 계수만 수정하여 더 큰 바닥을 자유롭게 만들 수 있다. 이러한 명령어의 효율성을 알려주기 위해 마지막에 학생에게 추가 질문을 하였다.

또한, 학생 g5의 명령어가 최소 코딩게임 적용 후 코드의 개수가 줄어들긴 하였으나, 더 짧게 줄일 방법이 있었으므로 질문을 통해 추상과 과정을 유도하였다. <명령어 1-2>에서 <명령어 1-3>으로 변화하는 과정에서는 꺽쇠 명령어를 이해하였으므로 수준 상승이 일어난 것을 확인할 수 있다. <명령어 1-4>, <명령어 1-5>로 변화하는 과정에서는 거북이의 경로가 중복되는 것을 알고 명령어를 더 효율적으로 작성하였다. 질문을 통해 학생 g5가 명령어를 수정하면서 코드의 개수가 점차 줄어들었는데, 이를 그림으로 표현하면 <표 Ⅳ-11>과 같다.

<표 Ⅳ-11> 문제 (1)에 대한 학생 g5의 명령어 변화과정

명령어 실행 결과	명령어 변화과정	핵심 사고 과정	코드 개수
	<명령어 1-1> goto(0,1,1) doit(5s 4l 4t 3r 3s 2l 2t 1r 1s)	치환을 사용하지 않고 거북이의 이동 경로에 따라 명령어를 씀	20
	<명령어 1-2> goto(0,1,1) X='[4l 4r t]' doit(5s 4X 4l)	반복되는 부분을 인지하여 치환을 사용함	13
	<명령어 1-3> goto(0,1,1) X='[4l] t' doit(5s 4X 4l)	꺽쇠 명령어를 이해하고 거북이의 이동 경로가 중복되지 않도록 수정함	11
	<명령어 1-4> goto(0,1,1) X='[4l] s' doit(4X 4l)	처음에 앞으로 5칸을 가지 않고 세로 1줄씩 만들면서 앞으로 나아가도록 경로를 수정함	8
	<명령어 1-5> goto(0,1,1) X='s [4l]' doit(5X)	왼쪽 4칸, 앞으로 1칸을 앞으로 1칸, 왼쪽 4칸으로 순서를 바꿔 실행 결과를 올바르게 수정함	7

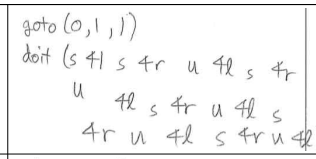
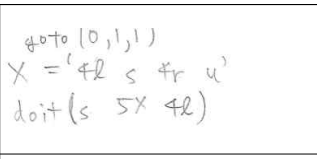
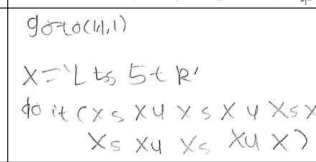
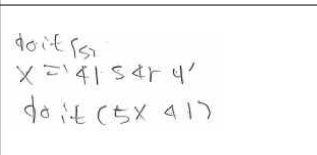
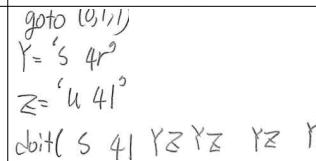
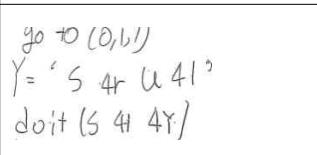
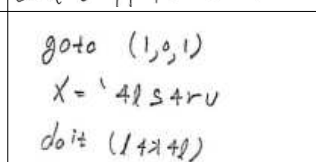
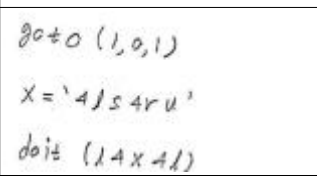
다. 최소 코딩게임 문제 (2) 분석 결과

최소 코딩게임 문제 (2)는 경회루 계단을 만드는 명령어를 코딩하는 문제이다. <표 Ⅳ-12>는 최소 코딩게임 문제 (2)에 대하여 교수전략을 적용하기 전과 적용한 후의 학생들의 답안이다.

<표 Ⅳ-12> 최소 코딩게임 문제 (2) 학생 답안

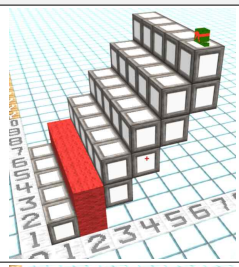
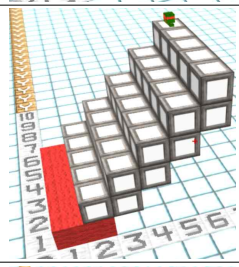
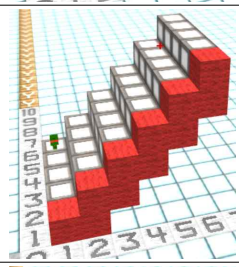
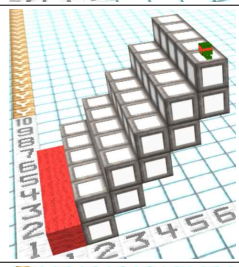
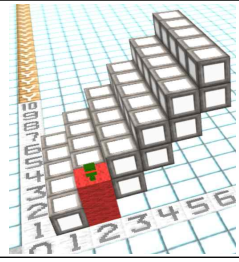
학생	최소 코딩게임 적용 전		최소 코딩게임 적용 후	
	코드	코드 개수 (유형)	코드	코드 개수 (유형)
g1	goto (1.0.1) doit(51) goto(2.0.1) doit(51) goto(2.0.2) doit(51) goto(3.0.2) doit(51) goto(3.0.3) doit(51) goto(4.0.3) doit(51) goto(4.0.4) doit(51) goto(5.0.4) doit(51) goto(5.0.5) doit(51) goto(6.0.5) doit(51) goto(6.0.6) doit(51)	44 (④)	goto (1.1.1) doit(51 5r) X='5 51 5r u 51 5r' doit(5X)	19 (①)
g2	goto(0,1,1) X='5 4r u 4l' doit(5 4l 5X)	13 (①)	goto(0,1,1) X='5 4r u 4l' doit(5 4l 5X)	13 (①)
g3	goto (0,1,0) X='5u[4L]' doit(6X)	8 (①)	goto(0,1,0) X='5u[4l]' doit(6X)	8 (①)
g4	goto (0,1,1) X='5[4L] u[4L] doit (5 [4L]) doit (5X)	14 (①)	goto(0,1,1) X='5[4l] u [4l]' doit (5 [4l]) doit(5X)	14 (①)
g5	goto(1,0,1) X='5 4r u 4l' doit (5l 5X)	12 (①)	goto(1,0,1) X='5 4r u 4l' doit (5l 5X)	12 (①)

g6	goto(0,1,1) X='2s 4s 4s4s4s4' doit(X) goto(0,2,1) doit(X) goto(0,3,1) doit(X) goto(0,4,1) doit(X) goto(0,5,1) doit(X).	26 (③)	goto(0,1,1) X='2s u s u s u s u' doit(X) goto(0,2,1) doit(X) goto(0,3,1) doit(X) goto(0,4,1) doit(X) goto(0,5,1) doit(v).	26 (③)
g7	X='[4l]s[4l]u' doit(15X5l)	13 (②)	X='[4l]s[4l]u' doit(15X4l)	13 (②)
g8	goto(1,0,1) doit(s\1s4r1u4l)1s4r1u4l)1s4r1u4l) 1s4r1u4l)1s4r1u4l) 1s4r1u4l)	44 (④)	goto(1,0,1) X='1s4r1u4l' doit(515X)	14 (①)
g9	cube(1,1,1) goto(1,1,1) X='4l s 4r u' doit(5X4l)	13 (②)	X='[4l]s[4l]u' doit(s5X4l)	13 (②)
g10	goto(0,1,1) goto(0,4,1) A='s u' doit(s 5A) doit(s 5A) goto(0,5,1) goto(0,2,1) doit(s 5A) doit(s 5A) goto(0,3,1) doit(s 5A)	27 (⑤)	goto(1,0,1) X='s[r u r u ru ru ru]' doit(L 5X)	16 (③)
g11	goto(1,1,1) cube(1,1,1) X='[s u su su su su]' doit(X1 X1 X1 X1 X1)	23 (③)	X='s [4l] u [4l]' doit(s [4l] 5X)	13 (①)
g12	goto(1,0,1) doit(51) X='s 4r u 4l' doit(5X)	13 (①)	cube(1,1,1) goto(1,1,1) X='4l s 4r u' doit(5X 4l)	13 (②)
g13	X='[4l]s[4l]u' doit(s 5X 4l)	16 (②)	X='[4l]s[4l]u' doit(s 5X 4l)	13 (②)

g14		35 (④)		13 (②)
g15		29 (④)		14 (②)
g16		19 (①)		13 (①)
g17		13 (②)		13 (②)

학생들의 답안을 반복되는 부분을 어떻게 인지하느냐에 따라 총 5가지 형태로 유형화하였다. 유형 ①은 계단에서 층이 다른 두 줄을 한 묶음으로 인지하여 반복되는 부분을 치환한 유형이고, 유형 ②는 계단에서 층이 같은 두 줄을 한 묶음으로 인지하여 반복되는 부분을 치환한 유형이며, 유형 ③은 계단의 맨 아래층부터 맨 위층까지를 한 묶음으로 인지하여 반복되는 부분을 치환한 유형이다. 유형 ④는 거북이가 왼쪽으로 4칸 가는 것을 한 줄로 인지하고 거북이의 경로를 표현한 유형이다. 반복되는 부분을 명확히 인지한 것이 아니라 거북이의 이동 경로대로 한 줄씩 표현하였다. 이때 치환을 사용하지 않고, 거북이가 왼쪽으로 4칸 간 뒤 오른쪽으로 4칸 가며 한 줄씩 이동하는 경우는 반복되는 부분을 인지하지 못하였으므로 유형 ④로 보았다. 유형 ⑤는 거북이의 경로를 앞으로 1칸 간 뒤 위로 1칸 가는 것으로 표현한 유형이다. 유형 ④와 마찬가지로 반복되는 부분을 명확히 인지하지 못하고 거북이의 이동 경로대로 표현하였다.

<표 Ⅳ-13> 최소 코딩게임 문제 (2) 분석 결과

유형	거북이의 경로	반복 인지 여부	최소 코드 (적용 후 평균)	적용 전 학생 수 (비율)	적용 후 학생 수 (비율)
①		O	8 (13.2)	6 (35.3)	8 (47.1)
②		O	13 (13.1)	4 (23.5)	7 (41.2)
③		O	16 (21)	2 (11.8)	2 (11.8)
④		X	29	4 (23.5)	0 (0)
⑤		X	27	1 (5.9)	0 (0)

즉, 유형 ①, ②, ③은 반복되는 부분을 인지한 경우이고, 유형 ④, ⑤는 반복되는 부분을 인지하지 못한 경우이다. <표 Ⅳ-13>을 보면, 유형 ①, ②, ③은 최소코드의 개수가 각각 8개, 13개, 16개이지만, 유형 ④, ⑤는 최소코드의 개수가 각각 29개, 27개로 유형 ①, ②, ③에 비해 길다. 반복되는 부분을 명확히 인지하여 치환하는 추상화 과정이 중요함을 알 수 있다.

최소 코딩게임 교수전략을 적용한 후, 코드의 개수가 감소한 비율이 47.1%, 변화 없는 비율이 5.29%, 증가한 비율이 0%였다. 특히, 유형 ①, ②, ③의 비율이 100%로 모든 학생이 반복되는 부분을 인지한 것으로 나타났다. 적용 전 유형 ④, ⑤의 학생이 적용 후 모두 유형 ①, ②, ③으로 변화하였다는 점으로 최소 코딩게임 교수전략이 학생들에게 효율적인 명령어를 작성하게 했음이 확인된다.

최소 코딩게임 적용 후 학생들의 핵심 사고 과정을 심층적으로 살펴보기 위해 코드의 개수가 가장 많이 변화하면서도 치환을 사용하지 않는 유형 ④에서 치환을 사용하는 유형 ①로 변화한 2인 1조 학생 g1에게 수업 시간 직후 심층 질문을 하였다.

연구자 : 문제 (2)에서 코드의 개수를 줄이려고 어떻게 한 거야?

g1 : goto 명령어를 없앴어요.

연구자 : goto 명령어를 없애면서 치환도 사용했네. 최소 코딩게임 전에는 치환을 왜 사용하지 않았니?

g1 : '5l'을 X로 치환하려 했는데, 굳이 하지 않아도 실행했을 때 계단이 잘 만들어져서 하지 않았어요.

연구자 : 최소 코딩게임 때 치환을 사용해서 명령어가 많이 짧아졌구나! 그런데, 자세히 보면 계단 모양이 조금 다르단다. 어느 부분이 다른지 계단의 너비를 잘 살펴보렴.

… (중략) …

g1 : '5l'을 '4l'로 바꿨어요. 선생님 말씀대로 doit도 한 번만 썼어요.

연구자 : 잘했어. 코드의 개수가 18개로 짧아졌네! 여기서 명령어를 더 짧게 쓸 수 있을까?

g1 : 음…. 힌트 주세요.

연구자 : '4l 4r'을 쓰면 거북이가 왼쪽으로 4칸을 갔다가 오른쪽으로 4칸을 가면서 쌓기나무를 중복으로 쌓게 돼. 수업 시간에 배운 명령어 중에서 거북이의 경로를 중복해서 쌓지 않기 위해 배웠던 명령어가 어떤 게 있었을까?

g1 : 꺽쇠 명령어요!

연구자 : 맞아, 잘 기억하네. 꺽쇠 명령어 '['와 ']'의 역할을 말해볼까?

… (중략) …

g1 : '4l 4r'을 '[4l]'로 바꿨어요.

연구자 : doit(4l 4r 5X)에서도 바꾸고 실행 결과를 확인해볼까?

g1 : doit([4l] 5X)로 바꿨는데 1칸이 비어요.

연구자 : 앞에 1칸을 채우려면 앞으로 1칸 가는 명령어를 쓰면 된단다. 거북이의 처음 위치를 앞으로 1칸 가려는 곳으로 바꿔서 비어 있는 곳을 채워볼까?

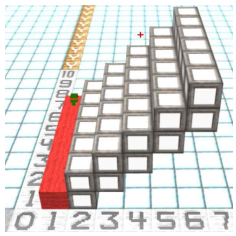
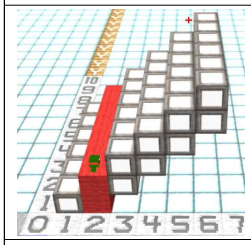
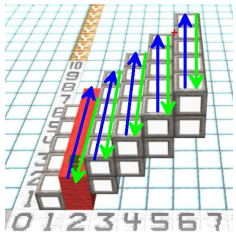
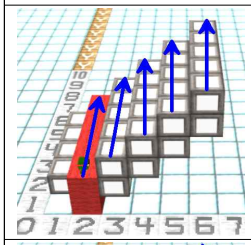
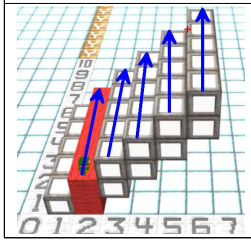
… (중략) …

g1 : 선생님, 했어요. 'doit([4l] 5X)'를 'doit(s [4l] 5X)'로 바꾸고, 'goto(1, 1, 1)'을 'goto(0, 1, 1)'로 바꿨더니 돼요!

연구자 : 아주 잘했어. 만약 계단을 많이 만들려면 어디를 고치면 될까?

학생 g1은 최소 코딩게임 전에는 치환을 사용하지 않고 거북이를 이동하여 한 줄씩 만들도록 하는 코드를 썼지만, 명령어를 짧게 쓰기 위해 계단의 층이 다른 두 줄을 한 묶음으로 치환을 사용하는 모습을 보였다. 심층 질문을 통해 학생 g1은 최소 코딩게임 전에 한 줄씩 만드는 '5l'도 치환할 수 있었으나 치환의 필요성을 느끼지 못하여 사용하지 않았다는 것을 확인하였다. 즉, 최소 코딩게임 후에 치환의 필요성을 느껴 치환을 사용하면서 수준 상승이 일어나는 모습을 보였다.

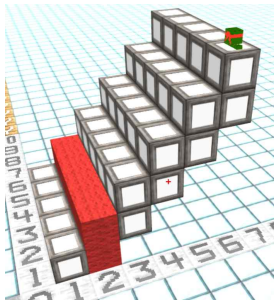
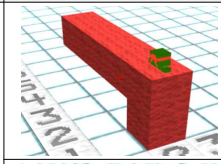
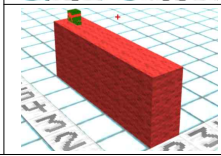
<표 Ⅳ-14> 문제 (2)에 대한 학생 g1의 명령어 변화과정

명령어 실행 결과	명령어 변화과정	핵심 사고 과정	코드 개수
	<명령어 2-1> goto(1,0,1) goto(3,0,3) doit(5l) doit(5l) goto(2,0,1) goto(4,0,3) goto(5,0,5) doit(5l) doit(5l) doit(5l) goto(2,0,2) goto(4,0,4) goto(6,0,5) doit(5l) doit(5l) doit(5l) goto(3,0,2) goto(5,0,4) goto(6,0,6) doit(5l) doit(5l) doit(5l)	반복되는 부분을 인지하지 못하여 치환을 사용하지 않고 거북이가 각각 한 줄씩 이동함	44
	<명령어 2-2> goto(1,1,1) doit(5l 5r) X='s 5l 5r u 5l 5r' doit(5X)	반복되는 부분을 인지하여 층이 다른 두 줄을 한 묶음으로 치환함	19
	<명령어 2-3> goto(1,1,1) X='s 4l 4r u 4l 4r' doit(4l 4r) doit(5X)	계단의 너비를 5에서 4로 수정함	19
	<명령어 2-4> goto(1,1,1) X='s 4l 4r u 4l 4r' doit(4l 4r 5X)	doit 명령어를 2번에서 1번 사용하도록 수정함	18
	<명령어 2-5> goto(1,1,1) X='s [4l] u [4l]' doit([4l] 5X)	꺽쇠 명령어로 거북이의 경로가 중복되지 않도록 수정함	12
	<명령어 2-6> goto(0,1,1) X='s [4l] u [4l]' doit(s [4l] 5X)	거북이가 앞으로 1칸 가는 것을 추가하여 실행 결과를 올바르게 수정함	13

또한, 학생 g1의 명령어가 최소 코딩게임 적용 후 코드의 개수가 줄어들긴 하였으나, 실행 결과가 제시한 계단의 모양과 약간 달랐으며 코드의 개수를 더 짧게 할 수 있었다. 이에 심층 질문을 통해 실행 결과를 올바르게 고치도록 하였고, 학생의 추상과 과정을 유도하였다. <명령어 2-1>에서 <명령어 2-2>로 변화하는 과정에서는 치환 문자를 사용했으므로 수준 상승이 일어난 것을 확인할 수 있다. <명령어 2-3>, <명령어 2-6>으로 변하는 과정에서는 잘못된 계단 모양을 올바르게 수정하는 과정을 거쳤다. <명령어 2-4>로 변하는 과정에서는 doit 명령어의 개수가 줄어들었고, <명령어 2-5>로 변화는 과정에서는 꺽쇠 명령어를 통해 거북이의 이동 경로를 중복되지 않게 바꾸면서 학생의 수준 상승이 일어나는 모습을 보였다. 질문을 통해 학생 g1이 명령어를 수정하면서 코드의 개수가 점차 줄어들었는데, 이를 정리하면 위의 <표 Ⅳ-14>와 같다.

그 외에도 학생 g3의 답안을 주목할 필요가 있다. 앞서 <표 Ⅳ-4>처럼 최소 코딩게임 문제 (2)를 구성할 때, 최소코드의 개수로 13개를 예상하였다. 그러나 학생 g3의 답안은 코드의 개수가 8개로 예상 최소코드의 개수와 차이가 있다. 그 이유는 학생 g3은 반복되는 부분을 두 줄이 아닌 시각적으로 보이는 부분만 인지했기 때문이다(<표 Ⅳ-15>). 후속 연구에서는 최소 코딩게임 문제를 제시할 때, 여러 방향에서 입체 구조물을 제시해야 하는 것으로 개선할 필요가 있다.

<표 Ⅳ-15> 문제 (2)에 대한 학생 g3의 명령어 분석 결과

명령어 실행 결과	반복 부분	코드	코드 개수
		goto(0,1,0) X='su [4l]' doit(6X)	8개
		goto(0,1,1) X='s [4l] u [4l]' doit(s [4l] 5X)	13개

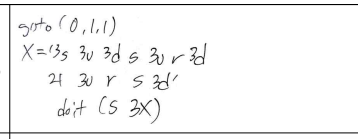
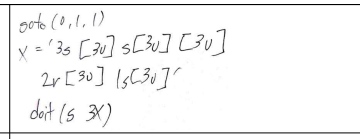
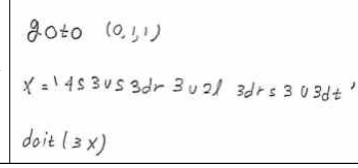
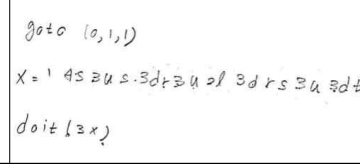
라. 최소 코딩게임 문제 (3) 분석 결과

최소 코딩게임 문제 (3)은 경회루 십자모양 기둥을 만드는 명령어를 코딩하는 문제이다. <표 Ⅳ-16>은 최소 코딩게임 문제 (3)에 대하여 교수전략을 적용하기 전과 적용한 후의 학생들의 답안이다.

<표 Ⅳ-16> 최소 코딩게임 문제 (3) 학생 답안

학생	최소 코딩게임 적용 전		최소 코딩게임 적용 후	
	코드	코드 개수 (유형)	코드	코드 개수 (유형)
g1	goto(0,1,1) doit(s) doit(3s [3u] s [3u] r [3u] 2l [3u] r s [3u]) doit(3s [3u] s [3u] r [3u] 2l [3u] r s [3u]) doit(3s [3u] s [3u] r [3u] 2l [3u] r s [3u])	59 (②)	goto(0,1,1) doit(s) X='3s [3u] s [3u] r [3u] 2l [3u] r s [3u]' doit(3x)	23 (②)
g2	goto(0,1,1) X='4s [3u] s [3u] r [3u] 2l [3u] r s [3u]' doit [3x)	22 (②)	goto(0,1,1) doit(s) X='3s [3u] s [3u] r [3u] 2l [3u] r s [3u]' doit(3x)	23 (②)
g3	goto(0,1,1) X='5s [3u][r 3u][l 3u][t 3u][s 3u]' doit (3x)	30 (③)	goto(0,1,1) X='5s [3u][r 3u][l 3u][t 3u][s 3u]' doit(3x)	19 (①)
g4	goto (0,1,1) doit(s) X='3s [3u] s [3u] s [3u]' doit(3x) goto (4,0,1) doit(s 3u) goto (1,0,1) doit (s 3u) goto (14,0,1) doit (s 3u) goto (4,2,1) doit (s 3u) goto (1,2,1) doit (s 3u) goto (14,2,1) doit (s 3u)	46 (⑤)	goto(0,1,1) X='5s [3u] [t 3u] [s 3u] [l 3u] [r 3u]' doit (3x)	19 (①)
g5	goto(0,0,1) doit (s) X='3s 3u 2s 3d 1t 1r 3u 3d 2l 3u 3d' goto (1,0,1) doit (x) goto (6,0,1) doit (x) goto(11,0,1) doit (x)	34 (④)	goto(0,1,1) X='3s 4u s 4d Rs 4u t l 4d r t 4u' doit (s X) goto (6,1,1) doit (x) goto (11,1,1) doit (X)	29 (④)

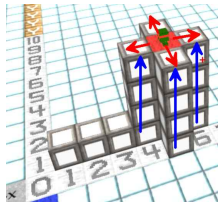
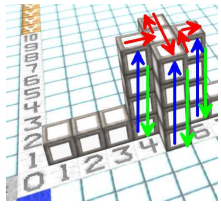
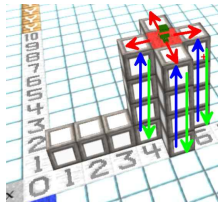
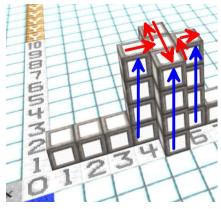
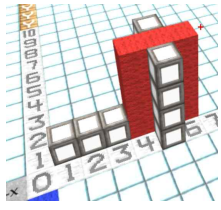
g6	goto(0,1,1) X='4s 3u 3d s 3u 3d r 3u 3d 2l 3u 3d r s 3u 3d' doit(3X)	32 (④)	goto(0,1,1) X = '3s [3u] s [3u] r [3u]' 2l [3u] r s [3u]' doit(3X)	23 (②)
g7	X = '3s [3u] s [3u] l [3u] 2r [3u] l s [3u] doit(s X XX)	24 (②)	X = '3s [3u] s [3u] l [3u] 2r [3u] l s [3u]' doit(s 3X)	23 (②)
g8	goto(0,1,1) X='3s[3us3dr3ul3dr3ul3dr]' doit(3X)	30 (③)	goto(0,1,1) X='3s[3us3dr3ul3dr3ul3dr]' doit(3X)	30 (③)
g9	cube (1,1,1) goto(1,1) X='3s 3u 3d s 3u l 3d 2r 3u l s 3d' doit(3X)	25 (④)	X='3s [3u] s [3u] [r 3u] [l 3u] s [3u]' doit (s 3X)	21 (①)
g10	goto(0,1,1) doit(s) X='3s 4u 4d s 4u 4d r 4u 4d 2l 4u 4d r s 4u 4d' doit(3X)	34 (②)	goto (0,1,1) doit (s) X = '3s [4u] s [4u] r [4u] 2l [4u] r s [4u]' doit (3X)	24 (②)
g11	goto(0,1,1) X='5s [3u] L [3u] L [3u] L [3u] L [3u] doit(3X)	24 (①)	X = 'L [3u]' Y='5s [3u]' doit(X X X X X X X X)	19 (①)
g12	goto(0,1,1) X='5[3u]' Y='2s' doit(6X2XLX2R6X2L5R XY2XL XOR 6X2L5R XY2XL XOR6X2L5R X)	50 (②)	goto(0,1,1) X='3s[3u] s[3u] L s[3u] 2R 2s[3u] 2l 5R s[3u]' doit(63X)	29 (②)
g13	X= [4u] [1s4u] L' doit (5s 4X 5s 4X 5s 4X)	21 (①)	X='[4u] [5 4u] L' doit (5s 4X 5s 4X 3s 4X)	20 (①)
g14	goto(0,1,1) X = '3u s 3d r 3u 2l 3d r s 3u 3d' doit (4s X 3s X 3sX)	29 (④)	goto(0,1,1) X = '3s [3u] s [3u] l [3u] 2r [3u] ls [3u]' do it (s 3X)	23 (②)
g15	doit(5) X='3s 3u 3d s 3u 3d L s 3u 3d 2l 3u 3d s r 3u 3d R' doit(3X)	35 (④)	doit(5) X='3s 3u 3d s 3u 3d L s3u. 3d 2l 3u.3d s r 3u .3d R' doit(3X).	35 (④)

g16		25 (④)		23 (②)
g17		25 (③)		25 (③)

최소 코딩게임 문제 (2)는 난이도 중으로 분류한 문제로 치환 문자의 적절한 사용 여부에 따라 명령어의 개수에 차이가 났다. 최소 코딩게임 문제 (3)은 난이도 상으로 분류한 문제로 치환 문자뿐만 아니라 꺽쇠 명령어의 적절한 사용 여부에 따라 명령어의 개수에 차이가 난다. 학생들은 이미 문제 (1), (2)를 해결하며 치환 문자의 필요성을 느껴 거의 모든 학생이 문제 (3)에서 치환 문자를 사용하였다. 이에 학생들의 답안을 유형화할 때, 꺽쇠 명령어의 사용 여부를 고려하였다. 또한, 꺽쇠 명령어를 사용하더라도 명령어가 가장 짧게 줄어들 수 있는 지점에서 꺽쇠 명령어를 사용해야 한다. 앞의 <표 Ⅳ-5>에서 보면, 십자모양 기둥은 5개의 일자모양 기둥이 합쳐진 것으로 구조적으로 중심축을 지니고 있다. 중심축에서 꺽쇠 명령어를 사용하는 것이 다른 곳에서 꺽쇠 명령어를 사용하는 것보다 명령어가 더 짧아진다. 꺽쇠 명령어를 사용하는 것뿐만 아니라 입체 구조물의 구조를 분석해야 한다는 것을 의미한다. 따라서 학생들의 답안을 십자모양 기둥의 중심축을 기준으로 꺽쇠 명령어를 어떻게 사용했느냐에 따라 5가지 형태로 유형화하였다.

유형 ①은 십자모양 기둥의 중심축을 기준으로 꺽쇠 명령어를 사용한 유형이고, 유형 ②는 꺽쇠 명령어는 사용하였으나 중심축은 파악하지 못한 유형이다. 유형 ③은 십자모양 기둥의 중심축을 파악하였으나, 꺽쇠 명령어는 사용하지 못한 유형이며, 유형 ④는 중심축도 파악하지 못하고 꺽쇠 명령어도 사용하지 못한 유형이다. 유형 ⑤는 반복되는 부분을 십자모양 기둥이 아닌 일자모양 기둥 3개를 합친 것으로 인지한 유형이다.

<표 Ⅳ-17> 최소 코딩게임 문제 (3) 분석 결과

유형	거북이의 경로	중심축 인지 여부	꺽쇠 사용 여부	최소 코드 (적용 후 평균)	적용 전 학생 수 (비율)	적용 후 학생 수 (비율)
①		O	O	19 (19.6)	2 (11.8)	5 (29.4)
②		X	O	23 (23.9)	5 (29.4)	8 (47.1)
③		O	X	25 (27.5)	3 (17.6)	2 (11.8)
④		X	X	25 (32)	6 (35.3)	2 (11.8)
⑤		X	O	46	1 (5.9)	0 (0)

즉, 유형 ①, ③은 십자모양 기둥의 중심축을 파악한 경우이고, 유형 ②, ④는 파악하지 못한 경우이다. 유형 ①, ②는 꺽쇠 명령어를 사용한 경우이고, 유형 ③, ④는 사용하지 못한 경우이다. <표 Ⅳ-17>을 보면, 유형 ①, ②는 최소코드의 개수가 각각 19개, 23개이지만, 유형 ③, ④는 최소코드의 개수가 각각 25개로 유형 ①, ②에 비해 길다. 결국 명령어를 짧게 쓰기 위해서는 꺽쇠 명령어를 적절히 사용할 수 있어야 하며, 이는 알고리즘의 선택 구조를 파악하는 수준으로 치환 문자로 반복 구조를 표현하는 수준보다 한 단계 더 수준이 상승해야 함을 의미한다.

최소 코딩게임 교수전략을 적용한 후, 코드의 개수가 감소한 학생 수가 13명, 변화 없는 학생 수가 3명, 증가한 학생 수가 1명이다. 증가한 학생 g2의 경우, 적용 전의 명령어가 제시된 십자모양 기둥과 약간 달랐기 때문에 올바르게 수정하는 과정에서 명령어가 증가하였다. 치환 문자는 교수전략 적용 전에도 94.1%의 비율로 거의 모든 학생이 사용하였다. 십자모양 기둥의 중심축을 파악하는 유형인 ①, ③의 비율이 교수전략 적용 전에는 29.4%였지만 적용 후에는 41.2%로 11.8%가 증가하였고, 꺽쇠 명령어를 사용하는 유형인 ①, ②의 비율이 교수전략 적용 전에는 41.2%였으나 적용 후에는 76.5%로 35.3%가 증가하였다. 특히, 최소 코딩게임 후에 꺽쇠 명령어의 사용 비율이 확연하게 증가했다는 점에서 치환 문자 사용의 반복 구조에서 꺽쇠 명령어 사용의 선택 구조로 수준 상승이 일어났음을 확인할 수 있다.

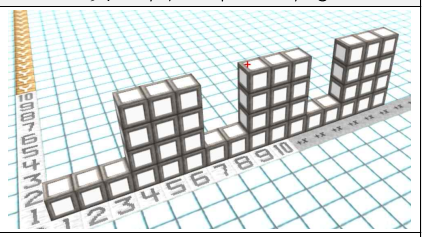
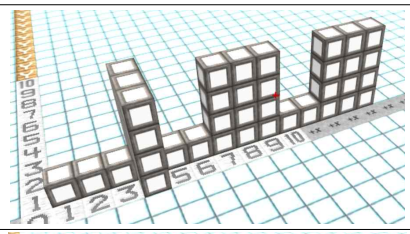
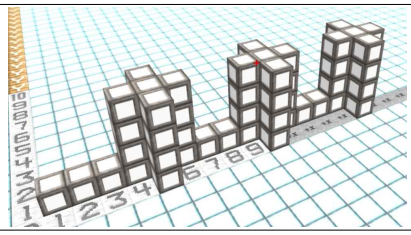
최소 코딩게임 적용 후 학생들의 핵심 사고 과정을 심층적으로 살펴보기 위해 코드의 개수가 가장 많이 변화하였으나, 유형은 유형 ②로 변하지 않는 2인 1조 학생 g12의 명령어 변화과정을 분석하였다. 이때, 터틀크래프트 온라인 홈페이지의 과제 자동 저장 기능을 활용하였다. 터틀크래프트 홈페이지에서는 [그림 Ⅳ-2]와 같이 과제 명령어 창을 생성할 수 있는데, 학생이 과제 명령어 창에 명령어를 입력하고 실행하면 시간 순서대로 명령어 실행 과정이 자동으로 저장된다. 교사는 이 기능을 활용하여 학생들의 명령어 변화과정을 확인할 수 있다.

[그림 Ⅳ-2] 터틀크래프트 온라인 홈페이지의 과제 자동 저장 기능

<표 Ⅳ-18>은 학생 g12가 십자모양 기둥을 만들기 위해 명령어를 실행하는 과정을 시간순으로 나열한 것이다. <명령어 3-1>을 보면 학생 g12는 십자모양과 관계없이 관통하고 있는 기둥들을 파악하고 있다. 이는 <표 Ⅳ-17>에서 유형 ⑤에 해당하는 것인데, 학생들이 유형 ①~④의 명령어를 썼다고 하더라도 처음 입체 구조물을 파악할 때는 전체 구조물을 관통하고 있는 부분을 파악하려는 경향이 있음을 알 수 있다. 이후 <명령어 3-2>에서는 반복되는 명령어를 인지하고 's[3u]'와 '2s'를 각각 X와 Y로 치환하고 있다. 최소 코딩게임 문제 (3)에서 대부분 학생이 치환을 사용하여 반복되는 부분을 찾아내어 복잡한 명령어를 간결하게 줄이는 추상화 과정이 일어났으므로 알고리즘의 반복 구조를 사용하는 수준까지 도달했다고 볼 수 있다. <명령어 3-3>과 <명령어 3-4>에서는 계속해서 명령어를 실행하고 수정하며, 제시된 십자모양 기둥을 만들어 나가는 과정이 확인된다. 즉, Papert의 Learning by Making 구성주의 교육전략이 학생들에게 이루어지고 있다. 최소 코딩게임 적용 후, 학생 g12가 수정한 <명령어 3-5>를 보면 명령어를 짧게 줄이기 위해 X와 Y

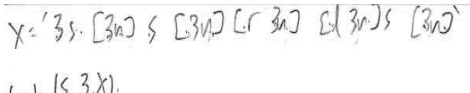
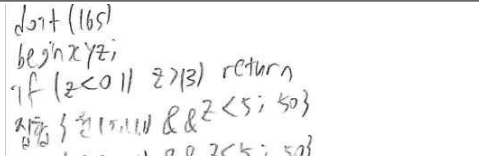
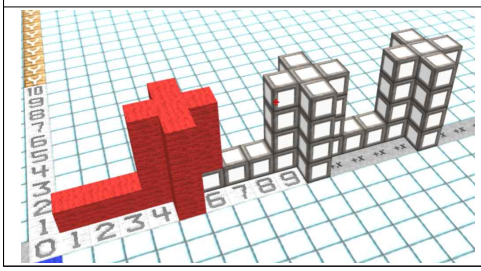
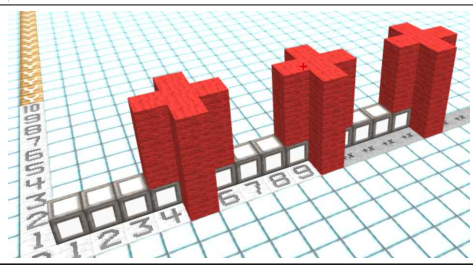
를 합쳤으나, ‘Ls’를 ‘l’로, ‘2Rs’를 ‘2r’로 고칠 수 있는 점은 알지 못하며, 십자모양 기둥의 중심축을 인지하는 수준까지는 도달하지 못하였다. 꺽쇠 명령어를 사용하는 선택 구조를 이해하는 것이 그보다 낮은 수준의 doit 명령어를 효율적으로 작성하는 것을 보장하지 않는다는 것이다.

<표 Ⅳ-18> 문제 (3)에 대한 학생 g12의 명령어 변화과정

명령어 변화과정	명령어 실행 결과 및 핵심 사고 과정
<명령어 3-1> 2022-06-13T13:35:15.358+09:00 goto(0,1,1) X='3s[3u] s[3u] s[3u]' doit(s 3x) 2022-06-13T14:37:58.329+09:00 goto(0,1,1) X='s[3u]'	
<명령어 3-2> 2022-06-13T13:41:01.742+09:00 goto(0,1,1) X='s[3u]' doit(3s3X2s3X2s3X) 2022-06-13T13:48:50.315+09:00 goto(0,1,1) X='s[3u]' Y='2s' doit(sY3XY3XY3X)	(최소 코딩게임 전) 일자모양 기둥 1개를 X로 치환 앞으로 1칸 가는 것을 Y로 치환
<명령어 3-3> 2022-06-13T13:49:08.185+09:00 goto(0,1,1) X='s[3u]' Y='2s' doit(sYXLX2RsX2LsRY3XY3X)	
<명령어 3-4> 2022-06-13T13:51:35.748+09:00 goto(0,1,1) X='s[3u]' Y='2s' doit(sY2XLX2RsX2LsRXY2XLX2RsX2LsRXY2XLX2RsX2LsRX)	
<명령어 3-5> 2022-06-13T14:37:58.329+09:00 goto(0,1,1) X='3s[3u]s[3u]Ls[3u]2R2s[3u]2LsRs[3u]' doit(s 3X)	(최소 코딩게임 후) X와 Y를 합쳐서 치환

그 외에도 학생 g9의 답안에 특이점이 발견되었다. 최소 코딩게임 문제 (3)을 제시했을 때, 학생들의 학습 상태는 doit, 치환, 꺽쇠 명령어와 더불어 집합 명령어까지 학습한 상태이다. 앞서 <표 Ⅲ-7>에서 언급한 것처럼 doit 명령어와 집합 명령어는 입체 구조물의 구조에 따라 더 편리한 명령어가 있다. 학생 g9는 십자모양 기둥을 만들 때, doit 명령어를 이용하는 것보다 집합 명령어를 이용하는 것을 더 편리하다고 느껴 교사에게 최소 코딩게임에서 집합 명령어 사용 가능 여부를 질문하였다. 학생 g9가 사용한 집합 명령어는 <표 Ⅳ-19>와 같다. doit 명령어를 사용할 때는 앞으로 3칸 가는 것과 십자모양 기둥 1개를 반복되는 부분으로 인지하고 X로 치환한다. 집합 명령어를 사용할 때는 앞으로 15칸을 doit 명령어로 만들고, 십자모양 기둥 각각을 집합 명령어를 사용한다. 집합 명령어가 더 효율적이라면 십자모양 기둥에서 집합 명령어를 사용하는 것을 생각해 볼 필요가 있다. 이에 후속 연구에서는 최소 코딩게임에서 집합 명령어를 활용하는 방안도 고려해볼 것을 제안한다.

<표 Ⅳ-19> 문제 (3)에 대한 학생 g9의 명령어 비교

doit 명령어	집합 명령어
	
	

3.2. 진로 탐색 및 수학과 코딩에 대한 인식의 변화

이번 장에서는 수학과 코딩을 융합한 코딩수학 교육과정을 자유학기제에 적용하고 난 후, 자유학기제의 목적 중 하나인 학생들의 진로 탐색에 대한 변화를 살펴보고자 한다. 구체적으로 다음과 같은 2가지 항목에 대해 분석한다.

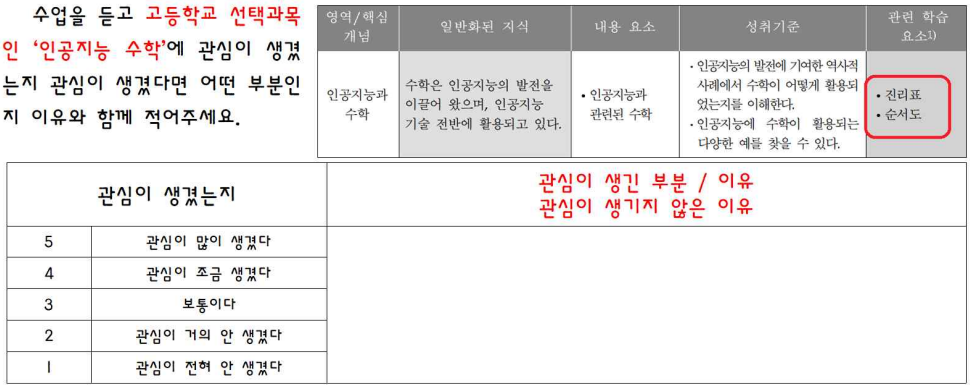
1) 고등학교 선택 과목 '인공지능 수학'에 대한 관심도
2) 수학과 코딩에 대한 인식

가. 진로 탐색의 변화

자유학기제의 목적은 학생들이 시험 부담에서 벗어나 자신의 소질과 적성에 맞는 꿈과 끼를 찾는 것이다. 김동심(2017)에 의하면 자유학기제 운영에 따른 진로 성숙도의 차이를 살펴본 결과, 진로 성숙도가 유의하게 증가한 것을 확인하였으며, 최윤정(2012)에 의하면 진로 교육이 강력하게 이루어진 학교일수록 학생의 진로 성숙도가 더욱 높게 증진되는 것을 확인하였다. 따라서 자유학기제 교육과정에서 학생들에게 유의미한 진로 경험을 제공해야 한다. 이에 본 연구에서는 자유학기제 교육과정에서 학생들이 코딩수학 수업을 듣고 난 후, 진로 탐색 태도에 어떤 변화가 있었는지 살펴보고자 한다.

한편, 교육부(2020)는 인공지능을 학교 교육에 적극 도입하기로 하며 2021년 2학기부터 고등학교에 '인공지능 수학' 과목을 선택할 수 있도록 하였다. 2022 개정 교육과정에서는 고교학점제가 전면적 시행에 따라 학생들의 '인공지능 수학' 과목을 선택할 기회가 더욱 자유로워졌다. 학생의 고등학교 선택 과목에 따라 학생의 대학 전공 분야 및 진로 분야의 방향이 결정되므로 자유학기제 교육과정에서 코딩수학 수업 후, '인공지능 수학' 선택 과목에 대한 학생들의 관심도 변화는 진로 탐색 태도에 대한 변화를 의미한다.

학생들에게 고등학교 선택 과목 ‘인공지능 수학’ 영역 중 ‘인공지능과 수학’ 영역의 성취기준 및 관련 학습 요소를 제시하고 관심도에 관한 설문을 실시하였다. [그림 Ⅳ-3]은 ‘인공지능 수학’에 관한 설문 문항이다. ⑤번 ‘관심이 많이 생겼다’, ④번 ‘관심이 조금 생겼다’, ③번 ‘보통이다’, ②번 ‘관심이 거의 안 생겼다’, ①번 ‘관심이 전혀 안 생겼다’에 해당하는 객관식 5점 척도 1문항과 관심이 생긴 부분 및 이유 또는 관심이 생기지 않은 이유를 묻는 서술식 1문항이다.


수업을 듣고 고등학교 선택과목인 ‘인공지능 수학’에 관심이 생겼는지 관심이 생겼다면 어떤 부분인지 이유와 함께 적어주세요.

영역/핵심 개념	일반화된 지식	내용 요소	성취기준	관련 학습 요소1)
인공지능과 수학	수학은 인공지능의 발전을 이끌어 왔으며, 인공지능 기술 전반에 활용되고 있다.	• 인공지능과 관련된 수학	• 인공지능의 발전에 기여한 역사적 사례에서 수학이 어떻게 활용되었는지를 이해한다. • 인공지능에 수학이 활용되는 다양한 예를 찾을 수 있다.	• 진리표 • 순서도

관심이 생겼는지		관심이 생긴 부분 / 이유 관심이 생기지 않은 이유
5	관심이 많이 생겼다	
4	관심이 조금 생겼다	
3	보통이다	
2	관심이 거의 안 생겼다	
1	관심이 전혀 안 생겼다	

[그림 Ⅳ-3] 고등학교 선택 과목 ‘인공지능 수학’에 관한 설문 문항

<표 Ⅳ-20>은 고등학교 선택 과목 ‘인공지능 수학’에 관한 객관식 설문 응답을 기수별로 분석한 것이다. ‘인공지능 수학’ 과목에 관심을 보인 학생은 ⑤번과 ④번에 응답한 학생으로 비율은 ⑤번 22.9%, ④번 42.9%, 총 65.8%에 해당한다. ‘인공지능 수학’ 과목에 관심을 보이지 않는 학생은 ②번과 ①번에 응답한 학생으로 7명의 학생이 이 항목에 응답하였다. 자유학기제 교육과정에서 코딩수학 수업을 듣고 난 후, 절반이 넘는 학생이 ‘인공지능 수학’에 대한 관심도가 높아진 것을 확인할 수 있다.

기수별로 살펴보면, 1기에서 ⑤번과 ④번에 응답한 학생으로 비율은 ⑤번 23.5%, ④번 52.9%, 총 76.4%이며, 2기에서 ⑤번과 ④번에 응답한 학생으로 비율은 ⑤번 22.2%, ④번 33.3%, 총 55.5%로 1기가 2기보다 ‘인공지능 수학’에 대한 관심도가 훨씬 높다.

<표 Ⅳ-20> 고등학교 선택 과목
'인공지능 수학'에 관한 객관식 설문 응답(기수별)

인원(명)	⑤	④	③	②	①	합계
1기	4	9	2	1	1	17
2기	4	6	3	3	2	18
전체	8	15	5	4	3	35
비율(%)	⑤	④	③	②	①	합계
1기	23.5	52.9	11.8	5.9	5.9	100
2기	22.2	33.3	16.7	16.7	11.1	100
전체	22.9	42.9	14.3	11.4	8.6	100.1

앞서 <표 Ⅳ-1>에서 1기와 2기의 학생 성별 비율을 보면, 1기는 남학생의 비율이 88.2%이고, 2기는 여학생의 비율이 66.7%인 특징을 지니고 있다. 이에 성별에 따른 특징을 파악하고자 고등학교 선택 과목 '인공지능 수학'에 관한 객관식 설문 응답을 성별로 분석하였다(<표 Ⅳ-21>).

<표 Ⅳ-21> 고등학교 선택 과목
'인공지능 수학'에 관한 객관식 설문 응답(성별)

인원(명)	⑤	④	③	②	①	합계
남자	5	10	3	1	2	21
여자	3	5	2	3	1	14
비율(%)	⑤	④	③	②	①	합계
남자	23.8	47.6	14.3	4.8	9.5	100
여자	21.4	35.7	14.3	21.4	7.1	99.9

성별로 살펴보면, 남학생은 ⑤번과 ④번에 응답한 비율이 각각 23.8%, 47.6%로 '인공지능 수학' 과목에 관심을 보인 비율이 총 71.4%이며, 여학생은 ⑤번과 ④번에 응답한 비율이 각각 21.4%, 35.7%로 '인공지능 수학' 과목에 관심을 보인 비율이 총 57.1%이다. 남학생과 여학생 모두 절반이 넘는 학생이 '인공지능 수학' 과목에 관심을 보였으나, 관심을 보인 남학생의 비율이 여학생의 비율보다 14.3%나 높다는 점에서 남학생이 코딩에 더 관심을 보였다고 해석할 수 있다. 그러나 연구 대상 인원이 적으므로 성별에 따른 관심도의 차이는 심도 있는 연구가 필요하다.

학생의 수준에 따라 '인공지능 수학' 과목에 관심도를 파악하기 위해 최소 코딩게임 문제의 결과를 활용하여 학생의 수준을 분석하였다. 알고리즘의 제어 구조에서 순차 구조보다 치환 문자를 사용하는 반복 구조가 수준이 높으며, 반복 구조보다 꺽쇠 명령어를 사용하는 선택 구조가 수준이 높다. 문제 (1), (2)에서는 치환 문자의 사용 여부로 수준을 구분하였으며, 문제 (3)에서는 꺽쇠 명령어의 사용 여부로 수준을 구분하였다. 또한, 최소 코딩게임의 규칙은 명령어를 가장 짧게 만드는 것이며, 명령어가 짧을수록 복잡성을 줄이는 추상화 과정이 일어난다. 추상화 과정은 컴퓨팅 사고력의 핵심 구성요소이며, 추상화 과정이 일어날수록 학생의 수준이 높아진다고 할 수 있으므로 최소코드의 작성 여부로 수준을 구분하였다(<표 Ⅳ-22>). 다만, 치환 문자나 꺽쇠 명령어를 사용했다고 하더라도 평균 코드 개수보다 3개 이상 많은 경우에는 하 수준으로 보았다.

<표 Ⅳ-22> 최소 코딩게임 문제의 수준 구분 형태

최소 코딩게임 문제	상 수준		중 수준		하 수준	
	최소 코드	치환 사용	최소 코드	치환 사용	최소 코드	치환 사용
문제 (1)	O	O	X	O	X	X
문제 (2)	O	O	X	O	X	X
문제 (3)	O	O	X	O	X	X

상 수준은 치환 문자를 사용하여 최소코드로 작성한 경우, 중 수준은 치환 문자는 사용하였으나 최소코드로 작성하지 못한 경우, 하 수준은 치환 문자도 사용하지 못하고 최소코드로도 작성하지 못한 경우이다.

<표 Ⅳ-23>은 최소 코딩게임 문제에 대한 학생 모둠별 응답 결과를 분석하여 수준을 정리한 것이다.

<표 Ⅳ-23> 최소 코딩게임 응답에 따른 학생 수준 분석 결과

모둠	문제 (1)		문제 (2)		문제 (3)	
	코드 개수	수준	코드 개수	수준	코드 개수	수준
g1	15	하	19	하	23	중
g2	7	상	13	상	23	중
g3	**7**	**상**	**8**	**상**	**19**	**상**
g4	7	상	14	중	19	상
g5	13	중	12	상	29	하
g6	**18**	**하**	**26**	**하**	**23**	**중**
g7	10	중	13	상	23	중
g8	7	상	14	중	30	하
g9	7	상	13	상	21	중
g10	16	하	16	중	24	중
g11	10	중	13	상	19	상
g12	10	중	13	상	29	하
g13	10	하	13	상	20	중
g14	16	하	13	상	23	중
g15	11	중	14	중	35	하
g16	10	중	13	상	23	중
g17	13	중	13	상	25	하
평균	10.88		14.19		23.94	

이 중에서 모든 문제에서 수준이 상으로 나온 모둠 g3과 문제 (1), (2)에서 수준이 하로 나온 모둠 g6을 대상으로 심층 인터뷰를 실시하였다. 모둠 g3과 g6은 모두 2인 1조로 모둠 g3의 학생을 s3, s4, 모둠 g6의 학생을 s6, s7로 지칭하겠다.

<표 Ⅳ-24>를 보면, 수준이 높은 학생 s3과 s4는 '인공지능 수학'에 대한 관심도가 있는 반면에, 수준이 낮은 학생 s6과 a7은 '인공지능 수학'에 대한 관심도가 보통이거나 없었다. 서술식 응답 내용을 살펴보면, 학생 s3은 코딩을 배우고 생각을 하며 퀴즈를 풀어 문제를 해결하는 것에 흥미를 느꼈고, 학생 s6은 어려운 부분에 관심이 생기지 않았으며, 학생 s7은 코딩에 자신이 없기에 관심이 생기지 않았다고 응답하였다. 즉, 수준이 높은 학생들은 최소 코딩게임 문제를 해결하는 데 흥미를 느낀 경향이 있으며, 수준이 낮은 학생들은 최소 코딩게임 문제를 해결하는 데 어려움을 느낀 경향이 있는 것으로 보인다.

<표 Ⅳ-24> '인공지능 수학'에 관한 서술식 설문 응답 (1)

학생	객관식 응답	서술식 응답 내용
s3	④ 관심이 조금 생겼다.	인공지능 수학이란 고등학교 진학과정에 조금 관심이 생긴 것 같다. 코딩에 대해 알아보며 배우고, 생각을 하며 퀴즈를 풀고 문제를 해결하는 것이 재밌었기 때문이다. 그리고 요즘에 코딩, 인공지능이 뜨리 주변에 많이 생기고 있고 활용되기 때문에 더욱더 관심이 가는 것 같다.
s4	⑤ 관심이 많이 생겼다.	아 그래도 인공지능 수학에 관심이 많고 좋아하는데 이번 수업을 통해 더 관심이 생겨 고등학교에서 인공지능 수학을 듣고싶다.
s6	③ 보통이다	새롭게 알게 되어 관심이 생긴 것도 있지만 잘 모르고 어려운 부분은 관심이 안 생겼다.
s7	① 관심이 전혀 안 생겼다.	생각해 보지 않은 개념인 데다 코딩에 자신이 없고 또 인공지능 코딩은 자신에 없기에 딱히 관심이 가지는 않았다.

심층 인터뷰는 '⑤ 관심이 많이 생겼다.'로 응답한 학생 s4와 '① 관심이 전혀 안 생겼다.'로 응답한 학생 s7를 대상으로 진행하였다.

<학생 s4 인터뷰>

연구자 : 원래부터 인공지능 수학에 관심이 많다고 했는데, 어떤 부분에서 관심이 많았니?

학생 s4 : 최근에 인공지능을 이용해서 만든 물건들이 나오고, 또 우리 생활을 편리하게 해줘서 관심이 있어요.

연구자 : 수업을 하면서 더 관심이 생겼다고 했는데, 수업의 어떤 부분을 할 때 관심이 더 생겼다고 생각하니?

학생 s4 : 음…. 코드를 맞춰서 결과물을 내는 것을 좋아해서 관심이 더 생긴 것 같아요.

연구자 : 그렇구나, 원래 게임 형태의 문제를 좋아하니?

학생 s4 : 네. 아, 그리고 평소에 마인크래프트 명령어를 사용하는 것과 커맨드 블록 다루는 것을 좋아했는데 수학 공부까지 해서 더 좋았어요.

연구자 : 수학을 좋아하는구나. 혹시 장래 희망이 무엇이니?

학생 s4 : 저 꿈이 과학자예요.

연구자 : 그러면 고등학교 가서 '인공지능 수학' 과목을 선택할 거야?

학생 s4 : 네, 재밌어요. 꿈이랑 관련도 있어서 선택할 거예요.

<학생 s7 인터뷰>

연구자 : 진로와 관련이 없다고 했는데, 혹시 진로가 무엇이니?

학생 s7 : 운동선수요.

연구자 : 운동을 좋아하는구나. 인공지능과 코딩에 자신이 없다고 했는데, 어떤 부분에서 그렇게 느꼈니?

학생 s7 : 그냥 원래 컴퓨터 만지는 걸 별로 안 좋아하고 코딩하는 게 어렵게 느껴져요. 수학도 별로 안 좋아해요.

연구자 : 그렇구나, 수업 시간에 배운 내용 중에서 처음에 배웠던 cube 명령어도 어려웠니?

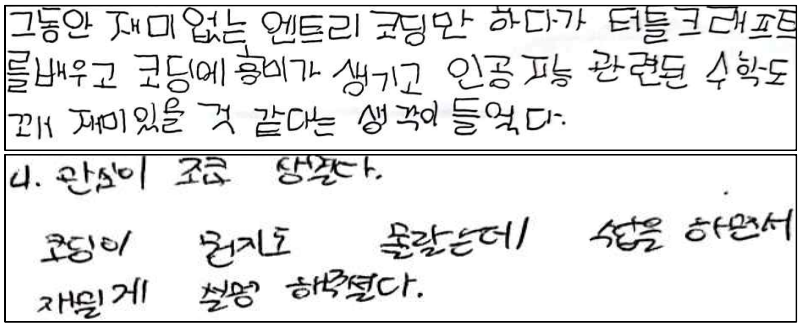
학생 s7 : 처음에는 좀 신기했는데, 딱히 배우고 싶지는 않아요.

연구자 : 나중에 어떤 부분에서 어려웠니?

학생 s7 : 대체로 생각보다 어렵고 나중에는 명령어가 많아져서 지루해서 어려웠어요.

학생 s3과 학생 s7과 인터뷰를 한 결과, 학생 s3은 진로가 코딩과 관련이 있었고, 본래 컴퓨터로 만드는 것에 흥미를 느꼈으며 수학을 좋아하는 경향이 있었다. 반면 학생 s7은 진로가 코딩과 관련이 거의 없었고, 본래 컴퓨터를 다루는 것과 수학을 어려워하는 경향이 있었다. 학생들의 진로 분야와 코딩 및 수학에 대한 선호도가 '인공지능 수학'에 대한 관심도에 영향을 미칠 수 있다는 점을 확인하였다.

반대로 [그림 Ⅳ-4]와 같이 코딩을 모르거나 코딩에 대한 선호도가 낮은 학생이 코딩수학 수업 후 '인공지능 수학'에 대한 관심도가 증가한 예도 확인할 수 있었다.



[그림 Ⅳ-4] '인공지능 수학'에 관한 서술식 설문 응답 (2)

이처럼 중학생의 경우, 진로 탐색 단계로서 자신의 관심 분야를 아직 모르는 경우도 있으므로 자유학기제에서는 학생에게 다양한 진로 경험을 제공하는 게 필요하다.

나. 수학과 코딩에 대한 인식 및 태도의 변화

앞의 장에서 수학과 코딩에 대한 선호도가 '인공지능 수학'의 관심도에 영향을 미칠 수 있다는 점을 확인하였다. '인공지능 수학'의 관심도는 진로 탐색 태도와도 관련되므로 수학과 코딩에 대한 인식 및 태도의 변화가 진로 탐색 태도의 변화와 관련이 있다. 따라서 이번 장에서는 자유학기제 교육과정에서 코딩수학 수업 전후로 학생들의 수학과 코딩에 대한 인식 및 태도 변화가 어떻게 변했는지 살펴보고자 한다.

'프로그래밍과 컴퓨팅 사고력에 대한 인식 변화와 과학 학습에 대한 태도 조사' 선행연구(황요한, 2016)에서는 1) 프로그램 활용에 대한 자신감, 2) 프로그래밍 언어 학습에 대한 인지 수준, 3) 컴퓨터 활용과 과학 학습과의 연계성, 4) 컴퓨팅 사고력에 대한 자신감, 5) 컴퓨팅을 통한 문제해결의 5가지 영역으로 나누어 설문조사를 실시하였다. 본 연구에서는 이러한 선행연구를 바탕으로 수학과 코딩에 대한 인식 및 태도 설문 문항을 구성하였다. 본 연구에서 실시하는 설문 영역은 4가지로 다음과 같다.

1) 코딩에 대한 자신감
2) 수학과 코딩에 대한 유용성
3) 수학과 코딩의 관련성
4) 문제에 대한 자신감 및 과제집착력

구체적인 설문 문항은 <표 Ⅳ-25>로 총 13개의 설문 문항으로 구성되어 있다. 1) 코딩에 대한 자신감은 1번부터 3번 문항까지, 2) 수학과 코딩에 대한 유용성은 4번부터 6번 문항까지, 3) 수학과 코딩의 관련성은 7번부터 9번 문항까지, 4) 문제에 대한 자신감 및 과제집착력은 10번부터 13번 문항까지다.

<표 Ⅳ-25> 수학과 코딩에 대한 인식 및 태도 설문 문항

번호	문항
1	나는 코딩을 배우면 이를 잘 이해하고 이용할 수 있다.
2	나는 코딩 학습이 다른 교과 학습보다 더 어렵다고 느낀다.
3	나는 코딩으로 문제를 해결하는 데 관심이 있다.
4	나는 코딩 학습이 4차 산업 시대에 필요하다고 느낀다.
5	나는 수학 학습이 4차 산업 시대에 필요하다고 느낀다.
6	나는 코딩이 수학 문제를 해결하는 데 도움이 된다고 생각한다.
7	나는 코딩 문제를 수학적으로 해결한 적이 있다.
8	나는 코딩이 수학과 관련성이 있다고 생각한다.
9	나는 코딩하는 데 수학 실력이 중요하다고 생각한다.
10	나는 주어진 문제를 푼 후에 풀이 과정을 다른 사람에게 논리적으로 설명할 수 있다.
11	나는 해결 전략이 바로 떠오르지 않거나 해결 방법이 다양한 문제를 도전해 볼 만하고 흥미 있는 문제라 생각한다.
12	나는 어렵고 도전적인 문제를 해결하는 것에 자신 있다.
13	나는 어렵고 도전적인 과제를 줬을 때 과제를 끝까지 해결하려는 경향이 있다.

설문은 ① ‘매우 그렇지 않다’, ② ‘그렇지 않다’, ③ ‘보통이다’, ④ ‘그렇다’, ⑤ ‘매우 그렇다’의 리커트 5점 척도로 응답하도록 구성하였다. 자유학기제 교육과정에서 코딩수학 수업 첫 차시 수업 시작 전과 코딩수학 수업 마지막 차시 수업 종료 후 같은 설문 문항으로 각각 설문조사를 실시하였다.

<표 Ⅳ-26> 수학과 코딩에 대한 인식 및 태도 설문 응답(기수별)

번호	1기			2기		
	사전	사후	차이	사전	사후	차이
1	3.5	3.6	+0.1	4.2	3.8	-0.4
2	2.6	2.7	+0.1	3.4	3.2	-0.2
3	3.6	3.5	-0.1	3.5	3.3	-0.2
4	4.4	4.7	+0.3	4.4	4.1	-0.3
5	4.4	4.4	-	4.3	4.3	-
6	3.9	3.7	-0.2	3.6	3.9	+0.3
7	**2.8**	**3.4**	**+0.6**	**2.5**	**3.3**	**+0.8**
8	4.2	4.3	+0.1	4.2	4.2	-
9	3.8	3.7	-0.1	4.1	3.7	-0.4
10	3.2	3.4	+0.2	3.7	3.3	-0.4
11	3.6	3.8	+0.2	3.7	3.7	-
12	3.2	3.1	-0.1	3.5	3.2	-0.3
13	3.6	3.4	-0.2	**4**	**3.4**	**-0.6**

<표 Ⅳ-26>은 수학과 코딩에 대한 인식 및 태도 설문 응답 평균을 기수별로 계산한 것이다. 대부분의 문항에서 사전 설문조사와 사후 설문조사의 결과가 큰 차이를 보이지 않았으나, 평균이 0.5 이상 차이 나는 문항도 2개 있었다.

7번 문항 '나는 코딩 문제를 수학적으로 해결한 적이 있다.'에서 1기는 2.8에서 3.4로 변화하여 0.6이 증가하였으며, 2기는 2.5에서 3.3으로 변화하여 0.8이 증가하였다. 많은 학생이 코딩과 수학이 관련성이 있다고 생각한 것으로 해석되며, 구체적으로 어떤 부분에서 코딩이 수학에 도움이 되는지는 뒤의 [그림 Ⅳ-5]에서 서술식 문항을 통해 분석하였다.

13번 문항 '나는 어렵고 도전적인 과제를 줬을 때 과제를 끝까지 해결하려는 경향이 있다.'라는 2기에서 4에서 3.4로 0.6 감소하였다. 1기에서

는 크게 차이가 나지 않았던 문항이 2기에서 차이가 나는 이유는 앞서 '인공지능 수학'에 대한 관심도에서 분석했듯이, 성별의 차이가 있을 것으로 예상할 수 있다. <표 Ⅳ-27>)는 수학과 코딩에 대한 인식 및 태도 설문 응답 평균을 성별로 계산한 것이다. 차이는 사전에서 사후로의 변화량을 의미한다.

<표 Ⅳ-27> 수학과 코딩에 대한 인식 및 태도 설문 응답(성별)

번호	여자			남자		
	사전	사후	차이	사전	사후	차이
1	3.4	3.6	+0.2	3.6	3.7	+0.1
2	**3.1**	**3.5**	+0.4	**2.5**	**2.7**	+0.2
3	2.9	3.3	+0.4	3.5	3.5	-
4	**3.7**	**4.2**	**+0.5**	4.3	4.5	+0.2
5	**3.7**	**4.4**	**+0.7**	4.2	4.3	+0.1
6	**3.1**	**4.0**	**+0.9**	3.7	3.7	-
7	**2.3**	**3.4**	**+1.1**	**2.5**	**3.2**	**+0.7**
8	**3.6**	**4.4**	**+0.8**	4.1	4.1	-
9	3.5	3.8	+0.3	3.8	3.7	-0.1
10	3.2	3.2	-	3.1	3.3	+0.2
11	**3.1**	**3.9**	**+0.8**	3.5	3.7	+0.2
12	2.9	3.0	+0.1	3.2	3.2	-
13	3.4	3.5	+0.1	3.6	3.3	-0.3

여학생은 수학과 코딩에 대한 인식 및 태도가 모든 문항이 전반적으로 증가한 반면 7번 문항을 제외하고는 남학생은 큰 차이를 보이지 않았다. 7번 문항은 기수별로 구분하여 계산하였을 때도 큰 차이를 보인 문항이다. 주목할 점은 여학생들이 4번 문항부터 8번 문항까지 0.5 이상 증가하였다는 점인데, 4번 문항부터 6번 문항까지는 수학과 코딩의 유용성, 7번

문항부터 8번 문항까지는 수학과 코딩의 관련성 영역이다. 따라서 여학생들은 수학과 코딩이 유용하다고 인식하는 정도와 수학과 코딩이 관련 있다고 인식하는 정도가 매우 증가한 것으로 해석된다. 남학생은 수학과 코딩의 유용성 및 관련성이 낮다고 인식한 것이 아니라 코딩수학 수업 전부터 여학생과 비교하여 이미 높은 수준으로 수학과 코딩이 유용하며 관련 있다고 인식하고 있다. 또한, 2번 '코딩 학습이 다른 교과 학습보다 더 어렵다고 느낀다.'라는 문항에서 여학생이 남학생에 비해 높은 것으로 보아 여학생이 코딩 학습을 어려워하는 경향이 있다는 점을 알 수 있다.

앞서 <표 Ⅳ-26>에서 7번 문항 '나는 코딩 문제를 수학적으로 해결한 적이 있다.'가 1기와 2기 모두 공통으로 증가하였는데, 코딩과 수학의 관련성에 대한 인식이 증가한 것으로 해석된다. 1기 사후 설문조사 실시 후, 가장 많은 변화가 일어난 7번 문항에 대해 구체적인 이유를 파악하기 위해 2기에서는 객관식 1문항과 서술식 1문항을 추가하였다. [그림 Ⅳ-5]와 같이 코딩수학 수업 후 코딩이 수학에 어떤 부분에서 도움이 되는지 또는 도움이 되지 않는 이유에 대한 설문 문항으로 실시하였다.

코딩수학 수업을 듣고 코딩이 수학에 도움이 되는지 도움이 된다면 어떤 부분인지 적어주세요.

코딩 수업이 수학에 도움이 된다		코딩이 수학에 도움이 되는 부분 / 코딩이 수학에 도움이 되지 않는 이유
5	도움이 많이 된다	
4	도움이 조금 된다	
3	보통이다	
2	도움이 거의 안 된다	
1	도움이 전혀 안 된다	

[그림 Ⅳ-5] 수학과 코딩의 관련성에 관한 설문 문항

<표 Ⅳ-28>은 '코딩이 수학에 도움이 된다'라고 생각하는지 5점 척도 객관식 문항에 대한 설문 응답 결과이다. '⑤ 도움이 많이 된다.'와 '④ 도움이 조금 된다.'라고 생각한 학생의 비율이 각각 33.3%, 38.9%로 도움이 된다고 생각한 학생의 비율이 총 72.2%였다. 대부분 학생이 '코딩이 수학에 도움이 된다'라고 생각하였다.

<표 Ⅳ-28> 수학과 코딩의 관련성에 관한 객관식 설문 응답

인원(명)	⑤	④	③	②	①	합계
2기	6	7	5	0	0	18
비율(%)	⑤	④	③	②	①	합계
2기	33.3	38.9	27.8	0	0	100

<표 Ⅳ-29>는 '코딩이 수학에 도움이 된다'라고 생각하는지 5점 척도 객관식 문항에 대한 설문 응답 결과이다. 편의상 1기 학생들을 s1, s2, …, s17로 지칭하고, 2기 학생들을 s18, s19, …, s35로 지칭한다.

<표 Ⅳ-29> 수학과 코딩의 관련성에 관한 서술식 설문 응답

학생	객관식 응답	서술식 응답 내용
s18	⑤	코딩 자체가 수학과 연관성이 있고 문자와 여러 코드, 식을 사용하면서 수학과 여러 접촉점이 있기 때문에 코딩 수업 자체는 수학을 하는데 많은 도움이 되었다. 그래서 (코딩이 수학에) 도움이 많이 된다고 생각한다
s19	⑤	코딩과 수학은 비슷하고 코딩이 수학에 도움이 생각보다 많이 되는 것 같다. 예를 들어 이 터틀 크래프트 에서는 (x, y, z) 좌표를 이용해 코딩을 하고 문제를 푼다. 또 마지막 단계에 그래프가 나오며 좌표가 나오는 걸로 안다
s20	③	그냥 블럭 위치만 정해서 옮기고 어린 걸처럼 초등학교 1학년도 할 수 있을 것 같았다
s21	④	수학(함수부분)에서 도움이 될 것 같다.
s22	⑤	좌표 평면 단원에서는 개념적으로 연관 되어서 도움이 많이 되고 그 외에 있어 크게 나눠진 거 같아 수학이라는 영역 [illegible] 않고 있어서 수학 실력이 향상되지 도움이 많이 된다.

s23	⑤	중학교 1학년 1학기 마지막단원 '좌표평면과 그래프'와 비슷해서 수학과 관련이 있다고 생각하고, X를 사용하여 코드를 쓰는 것도 수학과 관련이 있었고 이를 통해 조금 더 쉽게 수학을 이해할 수 있었다.
s24	③	수학에 도움이 되는 부분은 움직임과 코딩에 쓰는 숫자들이다
s25	⑤	공간에 대해 더 잘 알수있고 어느정도 연계이 된다.
s26	④	코딩을 이용하여 새롭게 수학을 할 수 있는 것 같다.
s27	④	코딩에 대해 알아보고 조금의 흥미가 생기게 되었다
s28	④	문제점이 -수학은 어렵지만 수학과 관련이 많이 되어있어서. 예습을 한거 같았다.
s29	③	기하학이나 그래프 같은게 나오는데 거기에 쓸수 있을거같다,
s30	④	코딩은 기하학과 관련이 있기 때문에 기하학은 수학 이므로 코딩은 수학에 도움이 된다.
s31	③	학교에서 배우고 있는것 또는 앞으로 배울것을 학습하게 되어 좋았고, 복습 및 예습을 동시에 재미있게 할수있어 좋았다. 수학에 흥미가 없고 컴퓨터에 흥미있는 아이에게 수학을 가르칠때 유용할것같다.
s32	④	④ 그래프를 활용해 코딩문자를 만들었다 그래프 덕분에 만드는게 훨 쉽다.
s33	③	움직이는 수와 좌표를 찍는 것을 통해 수학적 개념이 사용되어 도움이 조금은 된다고 할 수 있으나, 그 이외 수학적 계산은 사용되지 않아 큰 도움까지는 되지 않는다고 생각한다.
s34	④	수학에 가끔 좌표나 그래프 등에는 코딩계식이 필요할때가 있다. 그리고 도움도 된다
s35	⑤	5/ 좌표평면을 평소에 어려워했는데 코딩수학 활동을 통해 좌표평면이랑 좀더 친해진것 같다,

학생들의 응답 중 주목할 점으로는 학생 s27이 코딩에 흥미가 생기게 되었다고 한 점과 학생 s31이 수학에 흥미가 없지만, 컴퓨터에 흥미 있는 학생에게 수학을 가르칠 때, 유용할 것이라고 한 점이 있다. 또한, 구체적으로 어떤 수학 내용이 코딩에 도움이 되었는지 파악하기 위해 학생들의 응답에서 수학 관련 용어가 언급되는 횟수를 분석하였다.

<표 Ⅳ-30> 수학과 코딩의 관련성에 관한 설문 응답 중
수학 관련 용어 언급 횟수

쌓기나무(블록)	1	좌표	3
식	1	그래프	5
문자	1	(x, y, z)	1
숫자	2	함수	1
움직임	2	좌표평면	3
치환	1	공간	1
X	1	기하	1

<표 Ⅳ-30>을 보면, 그래프가 5회, 좌표와 좌표평면이 3회, 숫자와 움직임이 2회로 가장 많이 언급되었다. 특히 그래프, 좌표, 좌표평면이 언급이 많이 된 이유는 터틀크래프트 코딩환경이 3차원 좌표계를 기반으로 하기 때문으로 해석된다.

또한, ‘공간’이나 ‘기하’라는 용어도 등장하였는데, 이는 고등학교 수학과 교육과정에 나오는 용어이다. 권오남(1997)에 의하면, 공간 능력은 수학의 성취도에 관계되는 제 능력 중 그 성 차이가 가장 크게 나타나는 것으로 알려져 있으며, 공간 능력에서의 성별 차이를 연구한 결과 유의미한 차이를 보였다. 따라서 공간에서 기하를 다루는 터틀크래프트 코딩환경이 여학생들에게 어려움을 느끼게 할 가능성이 있으므로 코딩 학습과 관련하여 성별의 차이는 조금 더 심도 있는 연구가 필요하다.

Ⅳ. 요약 및 결론

1. 연구의 요약

4차 산업 시대가 도래함에 따라 전 세계적으로 SW 교육의 중요성이 대두되고 있다. 교육부(2015a)에서는 SW 교육의 중요성과 더불어 2015 개정 교육과정에서 추구하는 미래의 인간상으로 '창의·융합형 인재'를 강조하였으며, 2022 개정 교육과정에서는 추구하는 인간상으로 '창의성을 갖춘 주도적인 사람'을 강조하였다. 여기서 창의와 혁신은 문제해결, 융합적 사고, 도전 의식을 갖춘 사람을 말한다(교육부, 2022a). 또한, 2022 개정 교육과정에서는 중학교 1학년 자유학기제에서 디지털 소양을 포함한 기초소양 함양을 강조하고 있다. 이처럼 창의·융합형 인재 및 창의성을 갖춘 주도적인 인재를 양성하기 위해서는 자유학기제에서 서로 다른 지식을 융합하는 교육과정을 개발할 필요성이 있다.

그러나 자유학기제에서는 평가 결과가 점수로 표기되는 지필평가 위주의 중간고사나 기말고사를 실시하지 않는다. 평가의 부재는 학생들의 학습 동기가 효과적으로 유발되지 못할 수 있다. 자유학기제에서 학생들의 학습 동기를 효과적으로 유발하기 위한 교수전략이 필요하다.

이에 본 연구에서는 수학과 코딩을 융합한 코딩수학 교육과정과 이를 효과적으로 운영하기 위한 최소 코딩게임 교수전략을 디자인하고 실제 자유학기제 수업에 적용하였다. 이를 위하여 다음과 같은 연구 문제를 설정하고 하여 그 결과를 분석하였다.

첫째, 3차원 좌표계 기반 코딩환경을 활용하여 수학과 코딩을 융합하는 자유학기제 교육과정을 어떤 원리로 디자인할 수 있는가?

둘째, 디자인된 교육과정을 현장에 적용할 때, 학생들의 학습 동기유발을 위해 개발된 최소 코딩게임 교수전략에 따른 학생들의 학습 변화과정은 어떠한가?

셋째, 수학과 코딩을 융합하는 자유학기제 교육과정 기반의 수업을 통해 학생들의 진로 탐색에 대한 태도와 수학 및 코딩에 대한 인식에 어떤 변화가 있었는가?

각 연구 문제에 대한 결과를 요약하면 다음과 같다.

첫 번째 연구 문제에 대하여 본 연구에서는 수학 교과와 정보 교과의 공통 요소를 바탕으로 코딩수학 교육과정의 주요 수업 내용 및 학습 과제를 디자인하였다.

정보 교과의 알고리즘은 순차 구조, 반복 구조, 선택 구조로 되어 있는데 이를 수학 교과의 문자와 계수, 치환 문자 등과 융합하였다. 구체적으로 순차 구조에서는 문자와 계수, 반복 구조에서는 치환 문자를 도입하였다. 3차원 좌표계 기반 코딩환경을 활용하여 선택 구조에서는 꺽쇠 명령어를 도입하였으며, 수학 교과의 좌표, 순서쌍 개념을 다루고 있다. 또한, 정보 교과의 변수, 진리표를 수학 교과의 변수와 함수, 집합의 논리와 융합하였다. 이러한 코딩수학 교육과정은 자유학기제 교육과정의 운영에 맞게 8차시로 구성하였다. 각 차시의 주요 수업 내용 및 활동 과제는 <표 Ⅲ-5>와 <표 Ⅲ-6>으로 요약하였으며, 구체적인 활동 과제 내용은 <부록 1>로 수록하였다.

수학과 코딩을 융합하는 코딩수학 교육과정의 디자인 원칙은 다음과 같다. 첫째, 수준의 차이를 극복하는 교육과정으로 구성한다. 둘째, 수준 상승을 지향하는 교육과정으로 구성한다. 코딩수학 교육과정은 정보 교과의 컴퓨팅 사고력과 수학 교과의 수학적 사고력을 신장시켜 학생의 수준을 상승시키는 것을 목표로 하고 있다. 이에 알고리즘의 순차 구조, 반복 구조, 선택 구조 3단계로 수준을 구분하였으며, 각 단계에서 수준이 다른 명령어를 다룬다. 코딩수학 교육과정에서는 입체 구조물인 경회루의 구성요소를 명령어로 제작하는 활동 과제를 통해 학생들이 명령어를 학습하도록 하고, 최종적으로는 제작하는 경회루의 구성요소가 누적되어 전체 경회루를 완성한다. 이 과정에서 수준의 차이를 극복하기 위하여 수준이 낮은 1단계에서 도입되는 명령어로도 최종적으로 완성하는 경회

루를 만들 수 있도록 하였다. 또한, 활동 과제마다 제시하는 경회루의 구성요소에 따라 그 구성요소를 만드는 데 효율적인 명령어가 존재하여 경회루의 구성요소를 제작하는 수준이 다른 명령어를 도입할 수 있다. 학생의 수준 상승을 지향하여 수준이 낮은 1단계부터 수준이 높은 3단계까지 점차 수준이 높은 명령어를 사용하도록 활동 과제를 제시한다.

두 번째 연구 문제에 대하여 본 연구에서는 최소 코딩게임 교수전략을 개발하고, 이를 자유학기제 코딩수학 교육과정에 적용하여 학생들의 학습 변화과정을 분석하였다. 최소 코딩게임 교수전략은 다음과 같은 세 가지 원리를 통해 효과적인 지도전략으로 제안한다.

첫째, 학생의 동기유발 전략이다. 최소 코딩게임 문제는 학생에게 명령어를 가장 짧게 쓰게 하는 게임 형태로 제시되어 학생에게 도전 의식을 느끼게 하고 학생들은 서로의 응답을 공유하며 피드백을 받는다. 이 과정에서 한국콘텐츠진흥원(2013)이 제시한 즐거움을 느낄 수 있는 게임의 핵심 요인인 도전 의식과 피드백이 제공되어 학생의 학습 동기를 유발할 수 있다. 둘째, 학생의 수준 상승 유도 전략이다. 최소 코딩게임은 학생에게 명령어를 가장 짧게 최소로 표현하라는 조건을 줌으로써 반복하는 부분을 발견하여 복잡성을 줄이는 추상화 과정을 유도한다. 추상화 과정은 컴퓨팅 사고력의 핵심 요소로서 학생의 컴퓨팅 사고력을 신장시켜 수준 상승을 유도할 수 있다. 셋째, 평가 및 피드백 제공 전략이다. 최소 코딩게임은 특정 입체 구조물을 만드는 명령어를 가장 짧게 코딩하도록 문제를 제시하고 이를 점수화하여 그 결과를 즉각적으로 알 수 있게 하여 평가의 기능을 제공한다. 또한, 학생들은 서로의 응답을 공유하며 스스로 효율적인 코드를 발견하게 되고, 공유된 효율적 코드를 통해서 자신의 코드를 반성하게 된다(정진환, 2020).

최소 코딩게임을 위한 최소 코딩게임 문제는 총 3문항으로 좋은 문제의 요소에 맞게 다음과 같은 원리로 설계되었다. 첫째, 코딩수학 교육과정에서 최종적으로 만드는 건축물인 경회루의 구성요소를 만드는 문제여야 한다. 둘째, 학습한 명령어를 활용하면 명령어를 효과적으로 줄일 수

있는 문제여야 한다. 셋째, 거북이가 다양한 경로로 움직일 수 있는 개방형 문제여야 한다. 넷째, 추상화 과정을 유도하도록 반복되는 부분이 다양한 방법으로 나타나는 문제여야 한다. 다섯째, 학생들이 즉각적으로 쉽게 해결할 수 없는 도전 의식을 일으키는 문제여야 한다. 이를 만족하는 최소 코딩게임 문제는 각각 경회루의 구성요소인 바닥, 계단, 십자모양 기둥을 제작하는 문제이며, 난이도는 하, 중, 상에 해당한다.

최소 코딩게임 교수전략을 자유학기제 코딩수학 교육과정에 적용했을 때, 다음과 같은 결과가 나왔다. 첫째, 교수전략 적용 전후로 절반이 넘는 학생에게서 명령어를 효과적으로 작성하는 과정이 나타났다. 문제 (1), (2), (3)에 대한 학생의 응답에서 각각 코드의 개수가 감소한 비율이 58.8%, 47.1%, 76.5%였다. 둘째, 교수전략 적용 후, 모든 학생에게서 반복되는 부분을 발견하여 명령어의 복잡성을 줄이는 추상화 과정이 나타났다. 문제 (1), (2), (3)에 대한 학생의 응답에서 치환 문자를 사용한 학생의 비율이 최소 코딩게임 교수전략 후 모두 100%였다. 셋째, 교수전략 적용 후, 순차 구조보다 수준이 높은 반복 구조와 선택 구조를 사용하는 학생의 비율이 증가했다. 문제 (1), (2), (3)에 대한 학생들의 응답을 각각 유형화하였을 때, 수준이 높은 명령어를 쓰는 유형에 해당하는 학생의 비율이 모두 증가했다. 이를 정리한 표가 각각 <표 Ⅳ-10>, <표 Ⅳ-13>, <표 Ⅳ-14>이다. 이러한 결과를 토대로 최소 코딩게임 교수전략을 학생의 동기를 유발하여 수준 상승을 유도하는 효과적인 교수전략으로 제안한다.

세 번째 연구 문제에 대하여 본 연구에서는 자유학기제에 코딩수학 교육과정과 최소 코딩게임 교수전략을 적용한 후, 진로 탐색 및 수학과 코딩에 대한 인식 변화를 살펴보았다.

구체적으로는 고등학교 선택 과목 '인공지능 수학'에 대한 관심도와 수학과 코딩에 대한 인식 항목에 대해 분석하였다. 코딩수학 수업 후, 고등학교 선택 과목 '인공지능 수학' 과목에 관심을 보인 학생은 총 65.8%로 3분의 2에 해당하는 학생이 관심을 보였다. 응답 결과를 성별로 분석한

결과, 남학생이 71.4%의 비율로 여학생이 57.1%의 비율로 관심을 보였으며, 차이가 14.3%로 남학생이 여학생이 비해 더 관심을 보였다. 최소 코딩게임 문제 결과에 따라 학생의 수준을 구분한 후, 수준이 높은 학생과 수준이 낮은 학생 2명에 대하여 일대일 면담을 실시한 결과, 자신의 진로 및 관심 분야에 따라 이미 '인공지능 수학' 관심도가 결정되는 경향이 있었다. 반대로 코딩을 모르거나 코딩에 대한 선호도가 낮은 학생이 코딩수학 수업 후, '인공지능 수학'에 대한 관심도가 증가한 예도 확인할 수 있었다.

수학과 코딩에 대한 인식 및 태도 변화를 살펴보기 위한 설문조사는 선행연구(황요한, 2016)를 바탕으로 제작되었으며, 코딩에 대한 자신감, 수학과 코딩에 대한 유용성, 수학과 코딩의 관련성, 문제에 대한 자신감 및 과제집착력 4가지 영역에 대하여 실시하였다. 설문은 리커트 5점 척도로 응답하도록 구성하였으며, 각 문항에 대하여 평균을 계산한 결과, 7번 문항 '나는 코딩 문제를 수학적으로 해결한 적이 있다.'가 유의미하게 증가하였고, 13번 문항 '나는 어렵고 도전적인 과제를 줬을 때 끝까지 해결하려는 경향이 있다.'가 유의미하게 감소하였다. 7번 문항과 관련하여 2기 학생들에게서 수학과 코딩의 관련성에 관해 묻는 설문을 추가로 실시하였다. '코딩이 수학에 도움이 된다.'라는 설문에 대하여 도움이 된다고 생각한 학생의 비율이 총 72.2%였으며, 그 이유에 대하여 그래프, 좌표, 좌표평면이 언급이 가장 많이 되었다.

2. 연구의 제한점 및 후속 연구를 위한 제언

본 연구는 적은 인원의 학생을 대상으로 수업을 진행하고 설문 결과를 분석하였기에 연구 결과를 일반화하여 적용하기에는 어려움이 있을 수 있다. 성별에 따른 차이도 성별이 아닌 다른 개인 특성으로 인한 차이일 수도 있으므로 후속 연구에서는 보다 정확한 연구 결과를 위해 개인 특성에 관한 사전 조사가 필요할 것이다. 또한, 2015 개정 교육과정에서 전면 도입된 자유학년제가 2022 개정 교육과정에서 중학교 1학년 자유학기제와 중학교 3학년 진로연계 학기로 구분되었다. 본 연구에서는 중학교 1학년에서 자유학년제가 실시되는 상태에서 연구가 진행되었기 때문에 자유학년이 자유학기와 진로연계 학기로 구분하여 운영될 때와 차이가 존재할 수 있다.

이에 본 연구에서는 이상의 논의를 바탕으로 후속 연구를 위해 다음과 같은 제언을 하고자 한다.

첫째, 중학교 3학년 진로연계 학기에서 중학교 1학년 자유학기에 실시한 코딩수학 교육과정을 확장할 수 있는 방안에 관한 연구가 필요하다. [그림 Ⅱ-1]에서 제시한 것처럼 자유학기와 진로연계 학기는 단절된 것이 아니라 1학년에서 운영된 자유학기를 바탕으로 3학년에서 진로연계 학기가 운영된다. 자유학기에서는 디지털 소양을 포함한 기초소양을 함양하고, 진로연계 학기에서는 이를 토대로 진로를 탐색하고 고교생활을 준비한다. 따라서 본 연구에서 디자인한 자유학기제 코딩수학 교육과정을 진로연계 학기에서 연계하는 방안이 필요하다. 구체적으로는 본 연구에서 진로 탐색에 대한 변화를 알아보기 위해 실시한 설문 항목인 고등학교 선택 과목 ‘인공지능 수학’을 활용할 수 있다. ‘인공지능 수학’에서 사용하는 코딩환경인 파이썬을 소개함으로써 학생들이 ‘인공지능 수학’ 과목에 대하여 이해할 기회와 파이썬을 간단하게 접해봄으로써 ‘인공지능 수학’ 과목과 연관된 진로를 탐색할 수 있는 체험 기회를 제공할 수 있을 것이다.

둘째, 코딩교육과 융합 교육이 강조됨에 따라 코딩수학 교육과정을 초등학교에서도 초기 도입할 수 있는 방안에 관한 연구가 필요하다. 2019년부터 초등학교에서는 '실과' 과목의 단원에 코딩교육을 도입하고 있으며, 2022 개정 교육과정에서는 실과 교과를 포함하여 학교 자율시간을 활용하여 SW 교육을 17시간에서 34시간을 확장하여 운영한다. 본 연구에서는 변수 개념의 포함 여부에 따라 기본 교육과정과 심화 교육과정의 2가지 형태로 교육과정을 디자인하였다. 변수 개념은 중학교 1학년에서 배우는 수학 개념으로 기본 교육과정에서는 변수 개념이 포함되지 않는다. 따라서 초등학생들도 코딩수학 기본 교육과정 수업 내용을 학습할 수 있으며, 코딩수학 교육과정에서 다루는 터틀크래프트 3차원 코딩환경은 명령어로 쌓기나무 쌓는 활동을 제공하는 것으로 초등학교 수학 교과의 '공간과 입체' 단원과 관련이 있다. 이에 초등학교에서 기본 교육과정을 활용하는 후속 연구를 제안하며, 이를 위해 본 연구에서는 5)터틀크래프트 '코딩수학' 홈페이지의 중등(자유학기) 코딩수학 게시판에 본 연구의 활동 과제 자료 및 학위논문 파일을 게시해 두었다.

셋째, SW 교육 및 코딩교육과 관련하여 성별의 차이에 관한 후속 연구가 필요하다. 선행연구에 의하면, SW 창의교육을 초등학교 6학년 학생들에게 실시하였을 때, 창의적 문제해결력 및 SW 관련 진로지향도의 향상은 여학생보다 남학생에게서 뚜렷하게 나타났다(김인호, 2019). 본 연구에서도 SW 교육인 코딩수학 수업을 중학교 1학년 학생들에게 실시하였을 때, 코딩과 관련된 과목에 대한 관심도는 여학생보다 남학생이 훨씬 높았다. 그러나 본 연구에서는 연구 대상이 되는 학생의 인원이 총 35명으로 소규모 인원으로 연구 결과를 일반화하여 적용하기에는 어려움이 있을 수 있다. 또한, 자유학기제 교육과정에서 개설되는 주제 선택 프로그램이 4개밖에 없었는데, 이는 학생들이 선택할 수 있는 프로그램이 적어 2기 수업 때, 코딩수학 수업에 관심이 없는 학생이 많이 포함되었을 가능성이 있다. 2기 학생 중 여학생의 비율이 높았다는 점을 고려하

5) 터틀크래프트 '코딩수학' 홈페이지 URL : codingmath.org

면, 여학생의 특성이 아닌 수학 또는 코딩에 대한 관심도가 낮은 학생의 특성이었을 가능성도 배제할 수 없다. 이에 SW 교육 및 코딩교육과 관련하여 개인의 특성을 사전 조사하여 성별의 차이에 관한 심도 있는 연구가 필요할 것이다.

끝으로 본 연구에서는 수학과 코딩을 융합하는 코딩수학 교육과정을 디자인하여 이를 자유학기제에 적용하였으며, 이를 효과적으로 운영하기 위한 최소 코딩게임 교수전략도 함께 개발하여 적용하였다. 이를 통해 본 연구에서 디자인한 융합 교육과정과 교수전략이 2022 개정 교육과정에서 강조되는 SW 교육 및 융합 교육에 도움이 되기를 바라며, 본 연구에서 활용된 3차원 좌표계 기반 코딩환경 터틀크래프트 온라인 홈페이지에 해당 논문을 게재한다. 4차 산업혁명 시대를 맞아 컴퓨팅 사고력 신장을 위한 SW 교육 및 융합 교육을 실현할 수 있는 다양한 교육과정이 초등학교, 중학교, 고등학교에 걸쳐 이어지기를 기대한다.

참 고 문 헌

강하람, 임채령, 조한혁(2021). 수학 교과와 정보 교과를 융합하는 코딩수학 교육과정 및 교육방법 연구. 한국수학교육학회, 60(4), 467-491.

경상남도교육청(2021). 스스로 찾고 즐겁게 배우며 함께 성장하는 2021 자유학기제.

곽아영(2011). 게임을 통한 영어 학습동기강화 방안연구. 경희대학교 석사학위 논문.

교육부(2006). 7차 교육과정 수학과. 교육부 고시 제2006-75호 [별책 8].

교육부(2015a). 2015 개정 교육과정 총론 및 각론 확정·발표. 교육부 보도자료(2015.09.23.).

교육부(2015b). 고등학교 정보과 교육과정. 교육부 고시 제2015-74호 [별책 10].

교육부(2015c). 중학교 자유학기제 시행 계획(안).

교육부(2015d). 중학교 정보과 교육과정. 교육부 고시 제2015-74호 [별책 10].

교육부(2017). 중1 자유학기제, 내년부터 희망하는 학교 약 1,500개교에서 시작. 교육부 보도자료(2017.11.06.).

교육부(2020). 인공지능, 학교 속으로!. 교육부 보도자료(2020.09.11.).

교육부(2022a). 2022 개정 교육과정 총론 주요 사항 발표. 교육부 보도자료(2021.11.24.)

교육부(2022b). 2022 개정 교육과정 총론 주요 사항(시안).

교육부(2022c). 2022 개정 교육과정 총론 주요 사항의 신·구 대비표

권오남, 박경미, 임형, 허라금(1997). 공간 능력에서의 성별 차이에 관한 연구. 수학교육논문집, 5, 71-88.

권오남(2018). 제4차 산업혁명과 미래의 수학교육. 교육광장, 67, 10-13.

금종혜, 고호경, 권오남 외(2018). 고등학교 수학 교육과정 내용 축소가 이공계 인재 양성에 미치는 영향 분석. 한림연구보고서, 125.

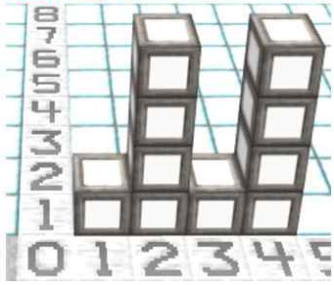
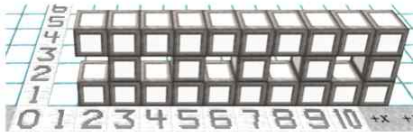
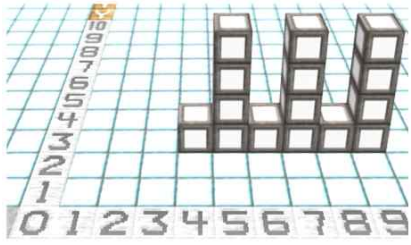
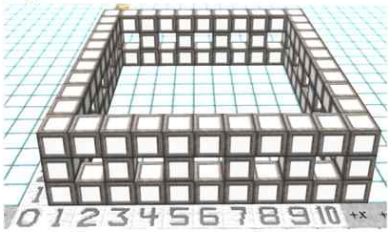
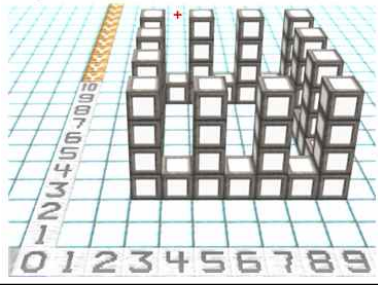
김동심(2017). 자유학기제 운영에 따른 교육성과 변화 분석: 진로성숙도, 인지적·정의적·사회적 핵심역량, 학교 만족도를 중심으로. 교육과정평가연구, 20(3), 101-121.
김윤민(2006). 좋은 문제에 대한 중등 수학 교사들의 인식 조사. 이화여자대학교 석사학위논문.
김인호, 유미현(2019). 소프트웨어(SW) 창의교육이 초등학생의 창의적 문제해결력 및 SW 관련 진로지향도에 미치는 영향 및 성별에 따른 차이 분석. 實科敎育硏究, 25(2), 151-177.
남충노, 김종우(2011). 컴퓨팅 사고력 기반 초등 수학의 교수 및 학습 활동 분석. 한국정보교육학회, 13(2), 325-334.
박남수(2018). 게임 기반 유아 소프트웨어교육에서 컴퓨팅 사고력에 영향을 미치는 요인규명. 이화여자대학교 박사학위 논문.
박종일, 박찬웅, 서효정, 염유식(2010). 한국 어린이-청소년 행복 지수 연구와 국제 비교. 한국사회학, 44(2), 121-154.
박현정(2008). 학습동기, 자아개념, 학업성취 간 관계의 집단 간 동등성 분석- PISA 2006을 중심으로. 교육평가연구, 21(3), 43-67.
백은미(2018). 자유학기제 운영실태에 관한 교사와 학생의 인식 비교 연구. 고려대학교 석사학위 논문.
신동조, 고상숙(2019). 수학교육에서 계산적 사고(Computational Thinking)의 의미 및 연구 동향 탐색. 수학교육, 58(4), 483-505.
신세리(2017). 다단계 인지진단평가 연구. 서울대학교 석사학위 논문.
신수범, 김철, 박남제, 김갑수, 성영훈, 정영식(2017). 정보과 교육과정에서 컴퓨팅 사고력과 연계한 디지털 소양 교육과정 프레임워크 개발. 정보교육학회논문지, 21(1), 115-126.
안창호, 이기쁨, 문석재(2018). 초보 학습자를 위한 엔트리 블록/텍스트 코딩 기반의 프로그래밍 학습 방법. 융복합지식학회논문지, 6(1), 127-134.
유가영(2017). 초등 수학에서 컴퓨팅 사고력 프로그램 개발 및 적용. 서

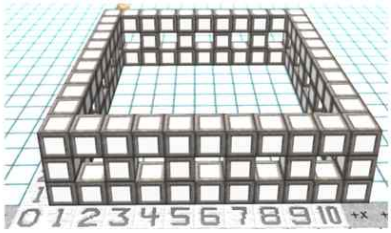
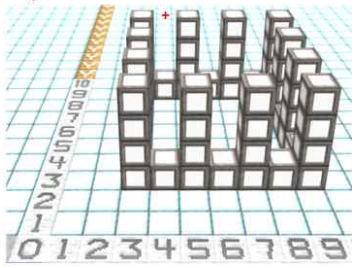
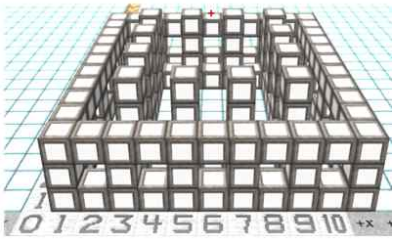
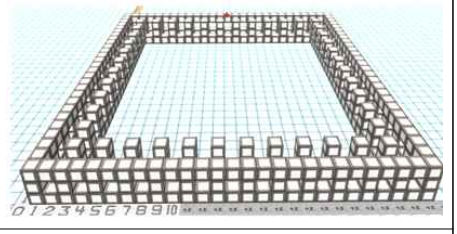
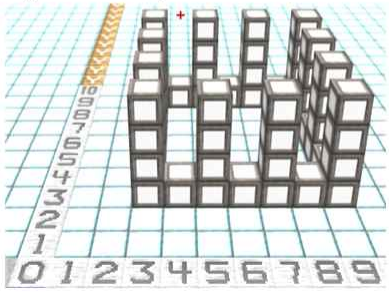
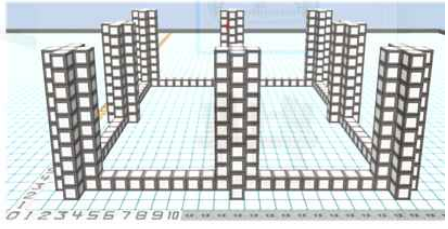
울교육대학교 석사학위 논문.
이명숙(2019). 마인크래프트 플랫폼을 이용한 소프트웨어교육 교수학습 모형. 디지털산업정보학회논문지, 15(3), 119-128.
이준열, 최부림, 김동재, 이정례 외(2020). 중학교 수학 2. 천재교육.
이환철(2019). 알지오매스(AlgeoMath)에 담긴 미래 수학교육의 방향 탐색. East Asian Math. J. 35(4), 387-406.
장경윤(2017). 수학 교과에서 계산적 사고(Computational Thinking)교육. 학교수학, 19(3), 553-570.
정영식, 강신천, 김민기 외(2020). 중학교 정보. 씨마스.
정인우(2020). 3차원 좌표계 기반 코딩환경을 활용한 수학 디자인 활동 연구. 서울대학교 석사학위 논문.
정진환(2015). Computational Thinking Game을 통한 패턴 일반화 수학 교육. 서울대학교 석사학위 논문.
정진환, 조한혁(2020). 코딩교육 명령문의 수학화: 대수 교육을 중심으로. 수학교육학연구, 30(1), 131-151.
조한혁, 송민호, 김화경(2006). The Qualitative Approach to Graphs of Functions in a Microworld. The SNU Journal of Education Research 15, 129-140.
주민정(2021). 알지오매스 블록 코딩을 활용한 자유학기제 수학과 주제 선택 활동 교수·학습 자료 개발. 한국교원대학교 석사학위 논문.
최윤정(2012). 중학생들의 진로교육 경험에 의한 진로교육 개입 유형 탐색 및 유형별 성과 차이. 진로교육연구, 25(2), 21-41.
최정원, 이은경, 김경훈, 이영준(2015). 2015 개정 정보 교과 교육과정에서 학습자의 컴퓨팅 사고력 평가 방안에 대한 제언. 한국컴퓨터교육학회 학술발표대회논문집, 19(2), 9-12.
한국콘텐츠진흥원(2013). 기능성게임 현황 활성화방안 연구 - KOCCA 연구보고서 13-11.
한국교육개발원(2013). 자유학기제 운영 프로그램 학생 수요조사 결과.

연구자료 CRM 2013-85.
한치근(2018). 컴퓨팅 사고력. 경기: 배움터.
황요한, 문공주, 박윤배(2016). 소프트웨어 활용 탐구 활동을 통한 고등학생의 프로그래밍과 컴퓨팅 사고력에 대한 인식 변화와 과학 학습에 대한 태도 조사: 스크래치와 피지컬 컴퓨팅 교구의 활용을 중심으로. 한국과학교육학회지, 36(2), 325-335.
ISTE & CSTA(2011). Computational Thinking in K-12 Education teacher resource 2nd Ed. Retrieved from https://www.iste.org/explore/computational-thinking/computational-thinking-all.
National Research Council. (2010). Report of a workshop on the scope and nature of computational thinking. Washington, DC: National Academies Press.
Papert, S. (1972). Teaching children to be mathematicians versus teaching about mathematics. International journal of mathematical education in science and technology, 3(3), 249-262.
Papert, S.(1980). Mindstorms: Children, Computers, and powerful ideas. New York: Basic Books.
Pintrich, P. & Schunk, D.(2002). Motivation in education: Theory, research, and applications. Upper Saddle River, NJ: Prentice Hall.
Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.
Wing, J. M. (2008). Computational thinking and thinking about computing. Royal Society of London Philosophical Transactions A, 366(1881), 3717-3726.

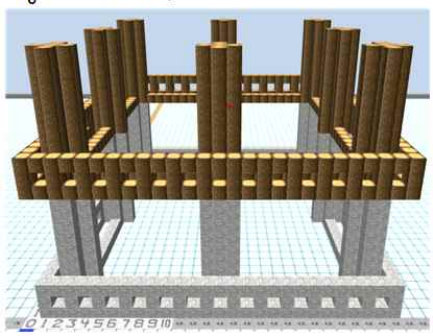
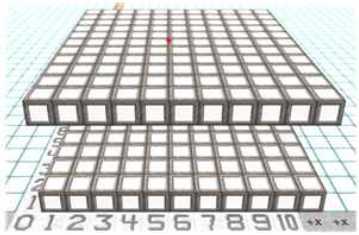
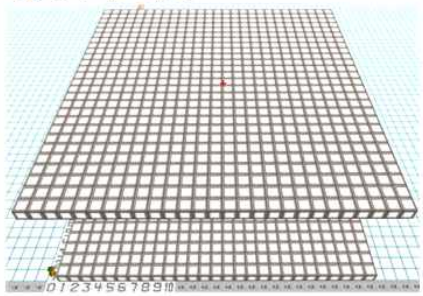
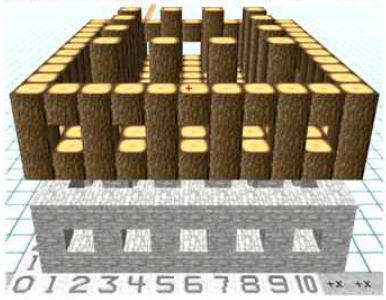
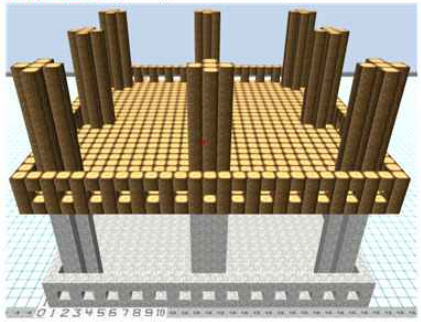
부 록

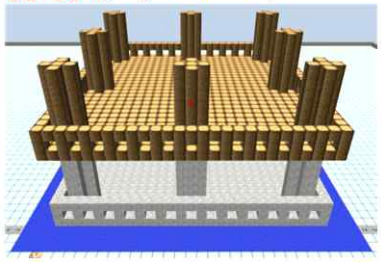
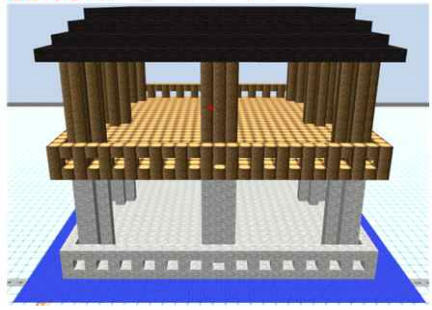
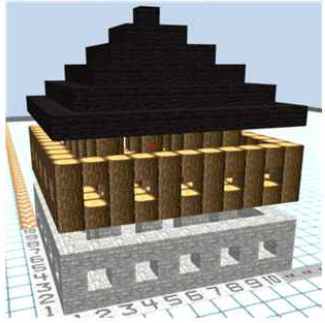
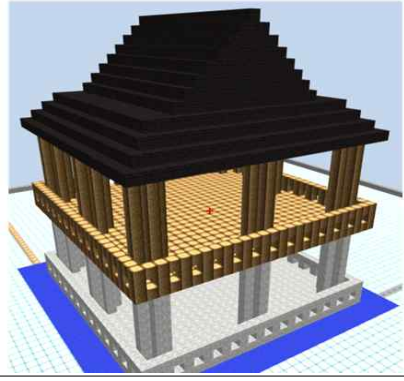
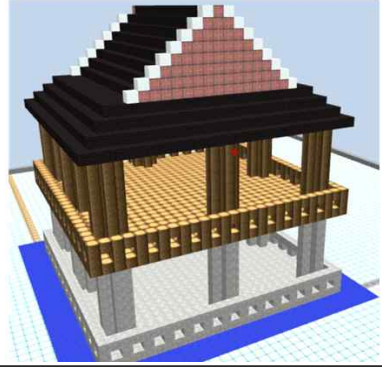
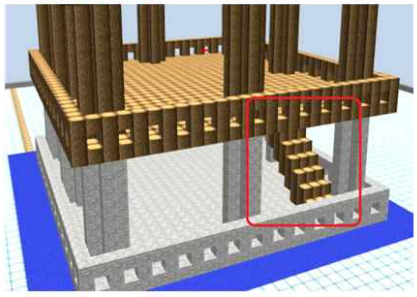
<부록 1> - 코딩수학 교육과정 활동 과제

차시	활동 과제	
1	과제 1번 경회루 난간 1개를 만드는 코드를 cube 명령어를 이용하여 써 보세요. 	과제 2번 경회루 기둥 2개를 만드는 코드를 cube 명령어를 이용하여 써 보세요. 
2	과제 1번 경회루 난간 5개를 만드는 코드를 goto 및 doit 명령어를 이용하여 써 보세요. 	과제 2번 경회루 기둥 3개를 만드는 코드를 goto 및 doit 명령어를 이용하여 써 보세요. 
3	과제 1번 경회루 난간 1바퀴 만드는 코드를 goto, doit 및 회전 명령어를 이용하여 써 보세요. 	과제 2번 경회루 기둥 1바퀴 만드는 코드를 goto, doit 및 회전 명령어를 이용하여 써 보세요. 

<table>
<tr>
<td rowspan="2">4</td>
<td>
과제 1번
경회루 난간 |바퀴 만드는 코드를
goto, doit, 회전 및 치환 명령어를
이용하여 짜 보세요.

</td>
<td>
과제 2번
경회루 기둥 |바퀴 만드는 코드를
goto, doit, 회전 및 치환 명령어를
이용하여 짜 보세요.

</td>
</tr>
<tr>
<td>
과제 3번 (기본)
경회루 난간 및 기둥 |바퀴 만드는 코드를
goto, doit, 회전 및 치환 명령어를
이용하여 짜 보세요.

</td>
<td>
과제 3번 (심화)
경회루 난간 및 기둥 |바퀴 만드는 코드를
goto, doit, 회전 및 치환 명령어를
이용하여 짜 보세요.

</td>
</tr>
<tr>
<td>5</td>
<td>
과제 1번 (기본)
경회루 난간 및 기둥 |바퀴 만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 명령어를
이용하여 짜 보세요.

</td>
<td>
과제 1번 (심화)
경회루 기둥 |바퀴 만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 명령어를
이용하여 짜 보세요.

</td>
</tr>
</table>

<table>
<tr>
<td>5</td>
<td>
과제 2번 (기본)
경회루 난간 및 기둥 아래층과 윗층 만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 명령어를
이용하여 써 보세요.

</td>
<td>
과제 2번 (심화)
경회루 난간 및 기둥 아래층과 윗층 만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 명령어를
이용하여 써 보세요.

</td>
</tr>
<tr>
<td rowspan="2">6</td>
<td>
과제 1번 (기본)
경회루 바닥 만드는 코드를
cubeset 명령어를
이용하여 써 보세요.

</td>
<td>
과제 1번 (심화)
경회루 바닥 만드는 코드를
집합 명령어를
이용하여 써 보세요.

</td>
</tr>
<tr>
<td>
과제 2번 (기본)
경회루 바닥, 난간 및 기둥만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 및
cubeset 명령어를 이용하여 써 보세요.

</td>
<td>
과제 2번 (심화)
경회루 바닥, 난간 및 기둥만드는 코드를
goto, doit, 회전, 치환 및 도돌이표 및
집합 명령어를 이용하여 써 보세요.

</td>
</tr>
</table>

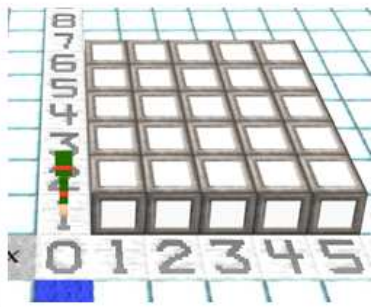
<table>
<tr>
<td>6</td>
<td>
과제 3번 (심화)
경회루 바닥, 난간 및 기둥, 연못 만드는 코드를 goto, doit, 회전, 치환 및 도돌이표 및 집합 명령어를 이용하여 써 보세요.

</td>
<td>
과제 4번 (심화)
경회루 바닥, 난간 및 기둥, 연못, 지붕하단 만드는 코드를 goto, doit, 회전, 치환 및 도돌이표 및 집합 명령어를 이용하여 써 보세요.

</td>
</tr>
<tr>
<td rowspan="2">7</td>
<td>
과제 1번 (기본)
경회루 바닥, 지붕, 난간 및 기둥 만드는 코드를 goto, doit, 회전, 치환, 도돌이표 및 cubeset 명령어를 이용하여 써 보세요.

</td>
<td>
과제 1번 (심화)
경회루 바닥, 난간 및 기둥, 연못, 지붕하단, 상단 만드는 코드를 goto, doit, 회전, 치환, 도돌이표 및 집합 명령어를 이용하여 써 보세요.

</td>
</tr>
<tr>
<td>
과제 2번 (심화)
경회루 바닥, 난간 및 기둥, 연못, 지붕하단, 상단 만드는 코드를 goto, doit, 회전, 치환, 도돌이표 및 집합 명령어를 이용하여 써 보세요.

</td>
<td>
과제 3번 (심화)
경회루 바닥, 난간 및 기둥, 연못, 지붕하단, 상단, 계단 만드는 코드를 goto, doit, 회전, 치환, 도돌이표 및 집합 명령어를 이용하여 써 보세요.

</td>
</tr>
</table>

<부록 2> - 최소 코딩게임 교수전략 전 활동지

바닥 만들기

다음 그림과 같은 **경회루의 바닥**을 만드는 코드를 써 보세요.

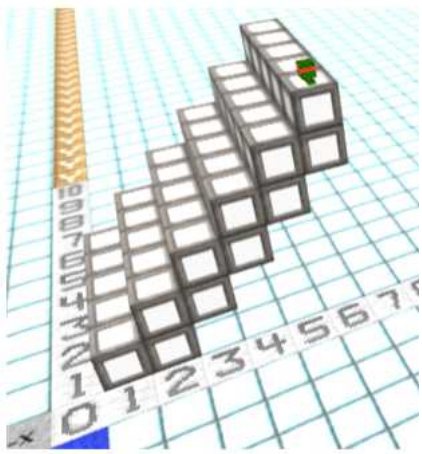
자신의 코드



계단 만들기

다음 그림과 같은 **경회루의 계단**을 만드는 코드를 써 보세요.

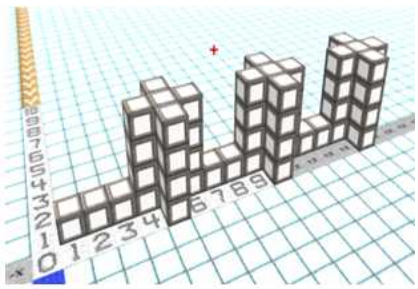
자신의 코드



기둥 만들기

다음 그림과 같은 **경회루의 십자기둥**을 만드는 코드를 써 보세요.

자신의 코드



<부록 3> - 최소 코딩게임 교수전략 후 활동지

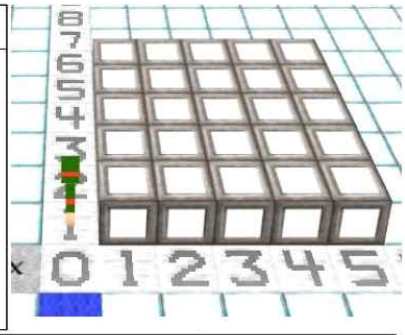
최소 코딩게임 - 바닥 만들기

다음 그림과 같은 **경희루 바닥**을 만드는 **가장 짧은 코드**를 써 보세요. 코드 개수가 가장 짧은 사람이 승리!!

< 개수 세는 법 >

1) 숫자 1개, 문자 1개당 글자의 개수 1개로 센다.
2) X=, [,], (,)와 쉼표, 따옴표는 개수로 세지 않는다.
3) X=' '안에 있는 글자는 개수로 세지 않는다.
4) doit, cube, goto 명령어는 1개로 센다.
5) cube와 goto 안에 있는 숫자는 개수로 세지 않는다.

반복되는 부분(치환한 부분)을 찾아 그려보세요

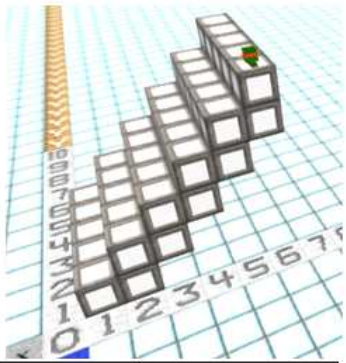


코드	코드의 개수

최소 코딩게임 - 계단 만들기

다음 그림과 같은 **경회루 계단**을 만드는 **가장 짧은 코드**를 써 보세요. 코드 개수가 가장 짧은 사람이 승리!!

< 개수 세는 법 >
1) 숫자 1개, 문자 1개당 글자의 개수 1개로 센다.
2) X=, [,], (,)와 쉼표, 따옴표는 개수로 세지 않는다.
3) X=' '안에 있는 글자는 개수로 세지 않는다.
4) doit, cube, goto 명령어는 1개로 센다.
5) cube와 goto 안에 있는 숫자는 개수로 세지 않는다.



코드	코드의 개수

최소 코딩게임 - 기둥 만들기

다음 그림과 같은 **경회루 기둥**을 만드는 **가장 짧은 코드**를 써 보세요. 코드 개수가 가장 짧은 사람이 승리!!

< 개수 세는 법 >
1) 숫자 1개, 문자 1개당 글자의 개수 1개로 센다.
2) X=, [,], (,)와 쉼표, 따옴표는 개수로 세지 않는다.
3) X=' '안에 있는 글자는 개수로 세지 않는다.
4) doit, cube, goto 명령어는 1개로 센다.
5) cube와 goto 안에 있는 숫자는 개수로 세지 않는다.

코드	코드의 개수

<부록 4> - 코딩과 수학에 대한 인식 및 태도 설문지

코딩교육 관련 설문조사

기본 인적사항 및 설문 결과 이용 동의

본 설문은 코딩교육과 관련하여 여러분의 의견을 수렴하여 효율적인 수업 운영 및 개선을 위한 목적으로 사용될 예정이며, 개인별 응답 내용은 모두 전산 처리되며 외부로 공개되지 않습니다. 기본 인적사항(성명) 및 설문 결과 이용에 따른 동의를 거부할 수 있습니다.

기본 인적사항 및 설문 결과 이용에 동의하십니까? 동의함 ( ) 동의하지 않음 ( )

기본 인적사항

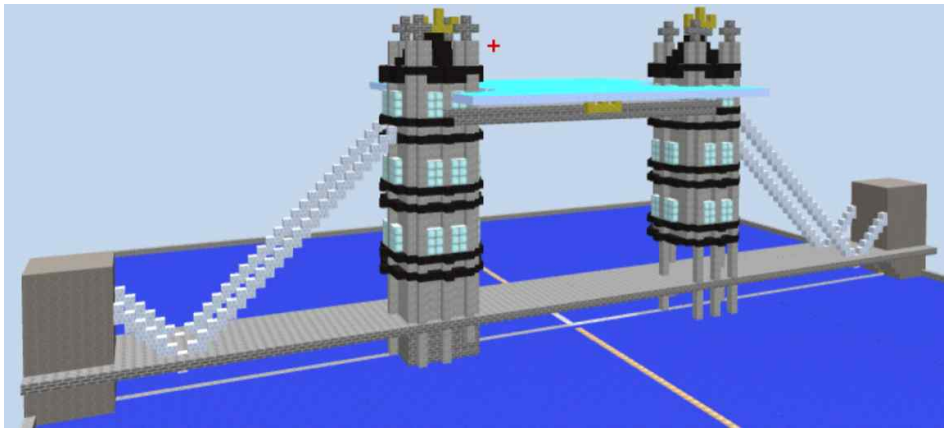
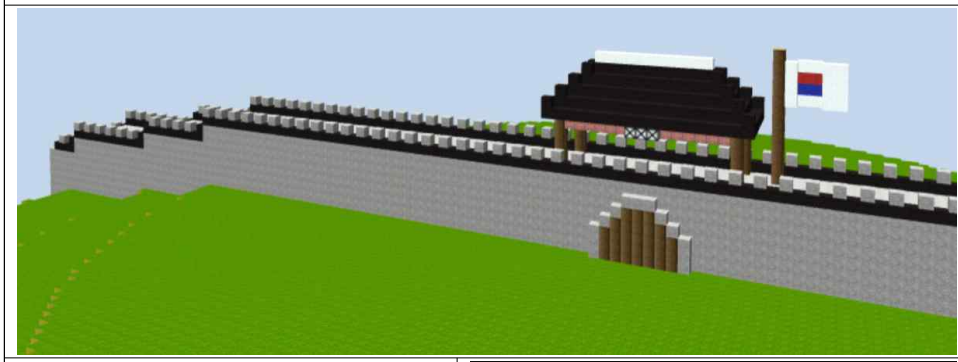
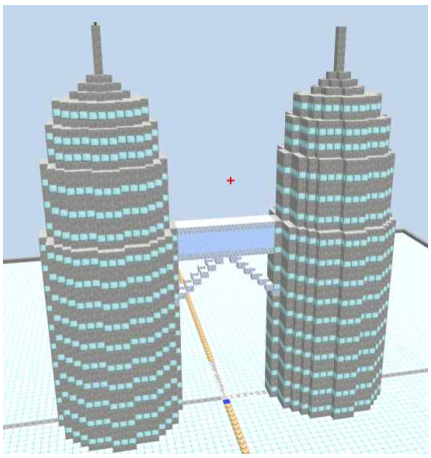
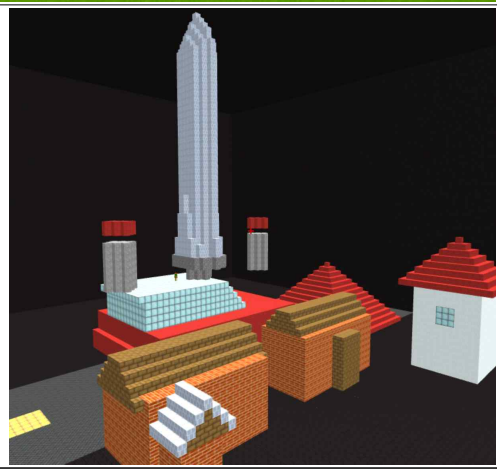
성 명	
성 별	남 ( ) 여 ( )
수학 실력	잘하지 못하는 편 ( ) 중간 정도하는 편 ( ) 잘하는 편 ( )
• 코딩을 다뤄 본 경험 (ex. 스크래치, 엔트리 등)	없다 ( ) 조금 있다 ( ) 많이 있다 ( )
• 다뤄본 코딩 프로그램	

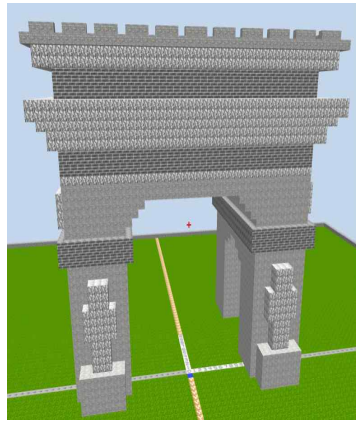
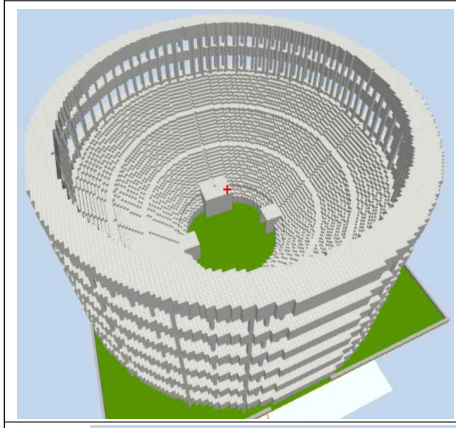
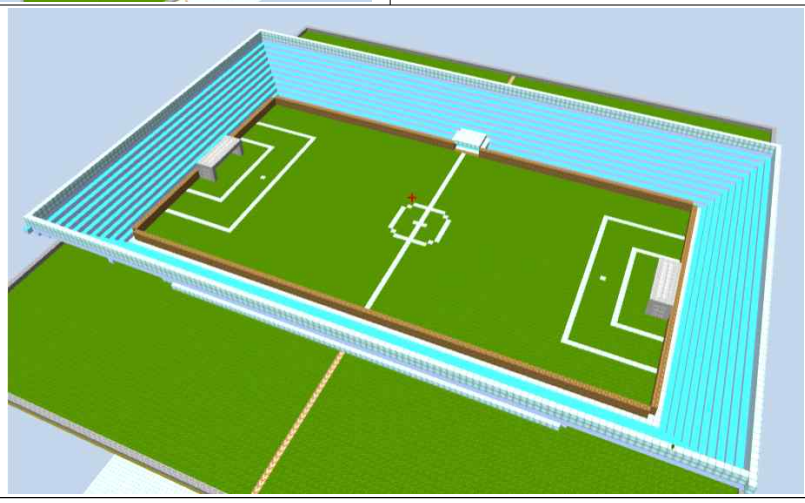
설 문 지

번호	문항	매우 그렇지 않다	그렇지 않다	보통	그렇다	매우 그렇다
1	나는 코딩을 배우면 이를 잘 이해하고 이용할 수 있다.	①	②	③	④	⑤
2	나는 코딩 학습이 다른 교과 학습보다 더 어렵다고 느낀다.	①	②	③	④	⑤
3	나는 코딩으로 문제를 해결하는 데 관심이 있다.	①	②	③	④	⑤
4	나는 코딩 학습이 4차 산업시대에 필요하다고 느낀다.	①	②	③	④	⑤
5	나는 수학 학습이 4차 산업시대에 필요하다고 느낀다.	①	②	③	④	⑤
6	나는 코딩이 수학 문제를 해결하는 데 도움이 된다고 생각한다.	①	②	③	④	⑤
7	나는 코딩 문제를 수학적으로 해결한 적이 있다.	①	②	③	④	⑤
8	나는 코딩이 수학과 관련성이 있다고 생각한다.	①	②	③	④	⑤
9	나는 코딩하는 데 수학 실력이 중요하다고 생각한다.	①	②	③	④	⑤
10	나는 주어진 문제를 푼 후에 풀이 과정을 다른 사람에게 논리적으로 설명할 수 있다.	①	②	③	④	⑤
11	나는 해결 전략이 바로 떠오르지 않거나 해결 방법이 다양한 문제를 도전해볼 만하고 흥미 있는 문제라 생각한다.	①	②	③	④	⑤
12	나는 어렵고 도전적인 문제를 해결하는 것에 자신 있다.	①	②	③	④	⑤
13	나는 어렵고 도전적인 과제가 주어졌을 때 과제를 끝까지 해결하려는 경향이 있다.	①	②	③	④	⑤

성실한 답변에 매우 감사드립니다.

<부록 5> - 최종과제 '나만의 건축물 만들기' 결과

<table>
<tr><th colspan="2">1기</th></tr>
<tr><td colspan="2">  </td></tr>
<tr><td colspan="2">  </td></tr>
<tr><td>  </td><td>  </td></tr>
</table>

27	
	
	
	

Abstract

A study on the design and application of a free semester system that converges mathematics and coding

Kang, Ha Ram
Department of Mathematics Education
The Graduate School
Seoul National University

The core of this study is to examine the results of designing and applying a free semester system curriculum that converges math and information coding so that students can find their dreams and talents and cultivate future competencies according to the purpose of the free semester system. This study is based on the need to converge different knowledge as the Ministry of Education emphasizes the image of a human being needed in the era of the 4th industrial revolution, in the 2015 revised curriculum, creative and convergence

talent, and in the 2020 revised curriculum, creative leaders. was carried out. In line with the introduction of the essential middle school information curriculum and high school artificial intelligence mathematics subjects, we aim to learn characters and substitutions, algorithms and flowcharts, variables and functions, geometry, and spatial figures, so that mathematics and coding can be fused with a three-dimensional coordinate system-based coding environment. We present a Turtlecraft learning environment with First, the design principle of the free semester system that converges mathematics and coding using Turtlecraft was discussed. Next, for the effective application of the designed curriculum, the minimum coding game teaching strategy was used and the learning change process of the students was analyzed accordingly. Finally, through a curriculum that converges mathematics and coding in the free semester system, students' attitudes toward career exploration and changes in their perceptions of mathematics and coding were examined.

For this purpose, in this study, 'coding math' class was conducted for 40 first-year students of S middle school located in Seoul. 'Mathematics of Coding' is a class conducted in the free semester system for middle school students, and deals with the fusion of coding and mathematics. When the minimum coding game teaching strategy was applied in the 'Coding Math' class, the students' commands before and after application were collected using the activity sheet to analyze the learning change process of the students. The command change process of the collected students was compared and the core thinking process of the students when using the command was analyzed. In addition, pre-study, post-study, and post-interview were conducted to examine students' attitudes toward career exploration and changes in their perceptions of mathematics

and coding through the 'Math coding' class.

As a result of the analysis, it was confirmed that when the minimum coding game teaching strategy was applied in the coding mathematics curriculum that converges mathematics and coding, both high-level students and low-level students were effective in motivating learning and inducing level rise. In addition, it was confirmed that there were desirable changes in students' career exploration and perception of mathematics and coding through the free semester system that converges mathematics and coding. This study is expected to suggest a curriculum that can nurture creative and convergence talents while achieving the purpose of the free semester system through research on the design and application of the free semester system that converges math and coding.

keywords : Free Semester System, Convergence Curriculum, Minimum Coding Game, Algorithm, Level Up, Learning Motivation

Student Number : 2018-23405