Journal of The Korean Association of Information Education Vol. 25, No. 6, December 2021, pp. 987-994

http://dx.doi.org/10.14352/jkaie.2021.25.6.987


# 컴퓨팅 사고력을 위한 프로그래밍 언어 교육과정
## : 라이트봇 게임과 고전 미로 게임으로 시작하기

전병우* · 신승기**

서울길원초등학교* · 서울교육대학교**

**요약**

컴퓨팅 사고력은 분석적 사고 능력으로, 누구에게나 또 어디에서나 필요한 능력이다. 실과 교과서에 제공되어 있는 기존의 컴퓨팅 사고력 개발 교육은 절차적 사고 능력 신장을 위한 언플러그드 활동에서 블록형 프로그래밍 언어로 이어진다. 다수의 언플러그드 활동은 놀이를 통한 순차적 사고 과정 연습에 초점을 두어 프로그래밍 언어에 필수적인 추상화나 자동화 과정에 대한 학습이 부족할 수 있다. 또 블록형 프로그래밍 언어에는 초등 교육과정에 소개되지 않는 좌표 평면 등의 개념이 등장하여 학생들이 블록형 프로그래밍 언어 자체에 부담감을 느끼게 만들기도 한다. 본 연구에서는 게임에 기반한 프로그래밍 언어 교육을 통해 초등학생의 컴퓨팅 사고력 개발을 위한 수업을 설계하였다. 수업 결과와 그 효과성은 비버챌린지를 통해 분석하였다. 분석 결과, 학생들의 컴퓨팅 사고력이 수업 전보다 향상되었음을 확인할 수 있었다.

키워드 : 컴퓨팅 사고력, 언플러그드 활동, 교육용 게임, 블록형 프로그래밍 언어

# Programming Language Curriculum for Computational Thinking
## : Starting with Lightbot hour and Classic maze

Bungwoo Jun* · Seungki Shin**

Gilwon Elementary school* · Seoul National University of Education**

Abstract

Computational Thinking is an analytical thinking ability that is necessary for everyone and everywhere. The existing Computational Thinking development education provided in Practical textbooks leads to block-based programming languages from unplugged activities. Many unplugged activities focus on practicing sequential order, which may lack the learning of abstractions or automation concepts. In block-based programming languages, concepts such as coordinate planes, which are not introduced in elementary school curriculum, appear, making students feel burdened by the block-based programming language itself. In this study, a curriculum was designed for elementary student's computational thinking through game-based programming language education. The results and their effectiveness were analyzed through the beaver challenge. As a result of analyzing the pre-test and post-test scores, it was confirmed that students' computational thinking skills improved.

Keywords : Computational Thinking, Unplugged activities, Educational game, Block-based programming language

---

본 논문은 (사)한국정보교육학회 2021년 하계학술대회에서 "비지도 학습을 위한 언플러그드 활동에 대한 연구"로 발표된 논문을 확장한 것임.
교신저자 : 신승기(서울교육대학교 컴퓨터교육과)
논문투고 : 2021-11-05
논문심사 : 2021-11-23
심사완료 : 2021-12-23

## 1. 서론

2022년은 대한민국 교육 역사에 있어 중요한 해다. 새로운 국가 수준 교육과정이 마련되고, 인공지능으로 대표되는 4차 산업을 위한 미래 인재 양성 노력이 시작되고 있다. 특히, 소프트웨어교육과 AI 교육을 토대로 초·중·고등학교 사이에 위계를 갖춘 컴퓨터 및 컴퓨팅 사고력 교육 도입의 필요성에 대한 목소리가 높아지고 있다.

7차 교육과정에서는 ICT교육을 토대로 초등학교 학생들에게 6년간 200시간 이상 컴퓨터 교육을 하기도 했다. 현재는 인공지능이 학생들의 생활 깊숙이 자리를 잡은 시대적 상황에서, 미래 인재 양성 필요성에 대한 목소리가 높아지고 있음에도 실과 교과 1개 단원에서 17시간을 교육하는 데 그치고 있다.

과거 김대중 정부에서는 IMF이후 새로운 도약을 준비하며 대한민국을 선진국으로 발전시키기 위해 미래학자인 앨빈 토플러에게 21세기 대한민국을 위한 보고서를 의뢰했다. Toffler(2001)는 프로그래밍 교육을 통해 컴퓨터 전문 인력을 양성함으로써 미래 대한민국의 발전을 선도할 수 있다는 의견을 제시한 바 있다. 새롭게 도입될 2022개정 교육과정을 통해 미래 인재를 양성하기 위해서는 2015개정 교육과정에서 제시된 핵심역량 중 하나인 지식정보처리 역량을 강화하고, 창의 융합형 인재 양성을 위해 정보 교육과정을 내실화할 필요가 있다.

본 연구에서는 2022개정 교육과정에 도입될 인공지능 교육의 실제적 교수학습 방법을 제시하기 위하여 게임 기반의 프로그래밍 언어 수업을 설계하고, 이를 통해 인공지능 및 소프트웨어 교육의 정착과 확산을 위한 방향을 제안하고자 하였다. Piaget의 인지 발달단계에 따른 학습자 분류에서 구체적 조작기에 해당하는 초등학생의 교육을 위해서는 충분한 시간의 구체적 조작 경험이 필수적이라 할 수 있다. 따라서 본 연구를 통해 프로그래밍 언어 교육에 필요한 추상화와 자동화 개념의 이해를 도와줄 구체적 조작 활동 중심의 게임 기반 프로그래밍 언어 수업 설계를 제시함으로써 개정 교육과정에 대비하고 학교 현장에서 적용 가능한 프로그래밍 언어 수업을 구성하여 그 효과성을 살펴보고자 하였다.

## 2. 이론적 배경

Wing(2008)은 컴퓨팅 사고력 (Computational Thinking)을 분석적인 사고라고 보았다. 분석적인 사고 능력은 누구에게나, 어디에서나 필요한 능력이라는 의견을 제시했다.

Brennan(2012)은 양방향으로 공유 가능한 블록형 프로그래밍 매체를 사용한 교육이 어린이들의 컴퓨팅 사고력 개발을 지원한다는 견해를 밝혔다.

Grover(2013)는 어린이들의 컴퓨팅 사고력 향상을 위해 비디오 게임 플랫폼을 사용하는 것이 가진 효과성에 비해, 관련된 연구의 수는 부족하다는 의견을 제시했다.

Weintrop(2017)은 음의 정수나 진법 등 초등 수학 교육과정의 범위를 넘어서는 개념이 등장하는 스크래치를 초등 정보교육에 도입하는 과정에서 발생하는 문제를 해결하기 위해, 스크래치 주니어 등의 대안적 프로그램에 대한 연구가 늘어나고 있다는 견해를 밝혔다. 그리고 블록형 프로그래밍 언어 수업은 학생들이 미래에 또 다른 컴퓨터 과목을 수강하고 싶게 만들 정도로 높은 수준의 흥미를 갖게 만든다는 의견도 제시했다.

Weintrop(2019)은 블록형 프로그래밍 언어 교육이 초보자들을 컴퓨터 교육에 입문하게 만드는 효과를 갖으며, 초등 고학년 학생들에게 적합한 매체라는 견해를 밝혔다. 그리고 블록형 프로그래밍 언어를 통해 아이들의 컴퓨팅 사고력을 발전시킬 최선의 방법을 찾는 것이 교육자들의 중요한 과제라는 의견을 피력했다.

Grover(2013)는 기초적인 컴퓨터 사용 경험은 '이용-변형-창조'라는 3단계를 거치며, 처음 컴퓨터를 배우는 아이들에게는 충분한 시간의 사용 경험 획득이 중요하다는 의견도 제시했다.

신승기(2018)는 핀란드의 정보교육에서는 프로그래밍 언어를 배우는 고학년이 되기 전에, 문제 해결 방법과 전략을 학습하는 충분한 시간의 놀이 과정이 정보 교육과정에 제시되고 있다는 분석을 하기도 했다.

전용주 등(2018)은 2015개정 교육과정에 제시된 sw 교육 성취기준 달성여부를 비버 챌린지를 통해 평가할 수 있다고 보았다. 비버챌린지는 신뢰도와 타당도가 높은 평가도구이며, CSTA의 K-12 컴퓨터 과학 표준을 반영하여 균형 있게 출제되었고, 이를 통해 컴퓨팅 사고력을 측정하고 촉진할 수 있다는 의견을 제시했다.

## 3. 연구방법

6학년 실과 교과를 통해 프로그래밍 언어를 접하는 46명의 학생들을 대상으로 연구를 진행하였다. 게임에 기반한 프로그래밍 언어 교육을 총 4차시에 걸쳐 실시하였으며, 수업 전과 후에 동일한 학생들을 대상으로 비버 챌린지 문항을 사용해 컴퓨팅 사고력을 평가하였다. 수업의 효과성은 대응표본 t-test를 통해 분석하였다.

### 3.1. 연구대상

서울 G초등학교에 재학 중인 6학년 학생 46명을 대상으로 본 연구를 진행하였다. <Table 1>은 연구 대상 학생들의 컴퓨터 교육 경험과 흥미 등에 대한 설문조사 결과이다. 80% 이상의 학생들이 컴퓨터 교육에 대한 흥미를 갖고 있었는데, 프로그래밍 언어를 배웠던 경험이 있는 학생의 비율은 10% 대에 불과했다. 이를 통해, 학생들이 흥미를 갖는 컴퓨터 교육 기회가 적절히 제공되지 못하고 있음을 알 수 있었다. 또 90%에 가까운 대다수의 학생들이 스크래치 등의 블록형 프로그래밍 언어를 부담스럽게 느끼고 있음을 확인할 수 있었다.

<Table 1> Percentage of student who said "Yes"

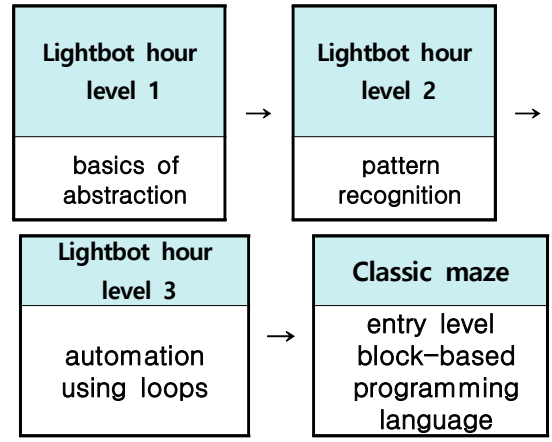
| Percentage | Question |
|---|---|
| 82.60% | Do you have any interest in learning computer? |
| 15.21% | Have you tried computer programming? |
| 10.86% | Was Scratch easy to use? |

### 3.2. 연구 목적 및 방법

본 연구는 초등학생들의 프로그래밍 언어 교육에 도움을 줄 교수학습 활동을 제공하는 데 그 목적이 있다. 현재 엔트리나 스크래치 등의 블록형 프로그래밍 언어가 6학년 실과 교과서에 소개되고 있다. 하지만, 프로그래밍 언어 도입 이전의 언플러그드 활동과 블록형 프로그래밍 언어 사이에는 간극이 있다.

학생들은 좌표 평면 등의 생소한 개념이 수많은 블록을 통해 갑작스럽게 제시되는 현재의 프로그래밍 언어 교육에 부담을 느끼는 경우가 있다. 그래서 간단한 게임에 기반한 활동을 통해 언플러그드 활동과 블록형 프로그래밍 언어 사이의 틈을 매우고자 하였다. 교육용 게임은 학생들의 심리적 부담을 덜어주고, 프로그래밍 언어 교육에 필요한 추상화 개념 등의 획득을 돕는 하나의 비계가 될 수 있다. 또 프로그래밍 언어 교육을 통한 컴퓨팅 사고력 신장에도 기여할 수 있다.

대면 수업과 비대면 수업이 혼재되어있는 학교 상황을 고려하여 가능하면 두 상황 모두에서 활용 가능한 교육용 게임을 선정하였다. 거리두기가 필요한 대면 교육에서는 학생들 각자의 스마트폰으로, 비대면 상황에서는 원격 수업 때와 마찬가지로 컴퓨터나 태블릿을 이용했다. 먼저 라이트봇 게임을 통해 순차, 반복, 순환 구조 등의 개념을 학습하고, 3-5개의 한정된 블록만을 사용하는 고전 미로 게임을 통해 순차, 반복, 순환의 개념을 블록형 언어로 표현해 본다. (Fig. 1) 과 같이 교육 활동이 총 4단계로 구성된 연구를 진행하였다.

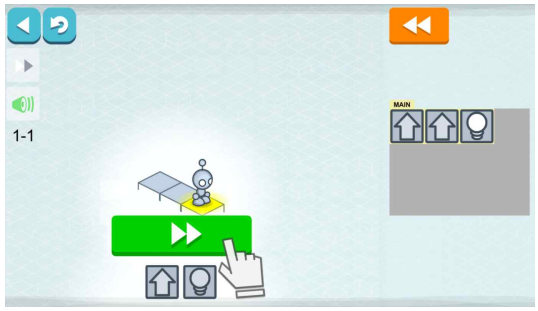



(Fig. 1) Instructional Model for Programming Language Using Educational Game

### 3.3. 재구성한 수업 설계

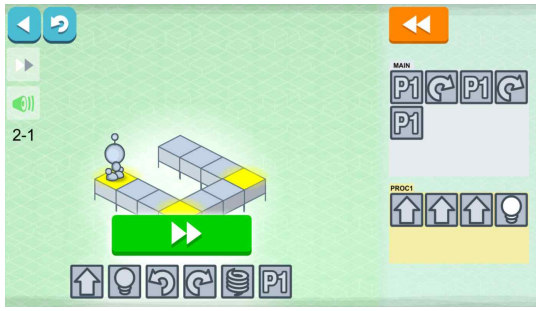
처음에는 라이트봇 1단계 게임 1-8이 제시된다. 먼저 (Fig. 2)의 MAIN 영역을 보면, 문제 해결에 사용해야 하는 명령어의 개수는 3개인데, 명령어를 입력할 수 있는 칸의 개수는 12개로, 사용할 수 있는 코드의 개수에 제약이 없음을 확인할 수 있다. 다음으로 앞으로 한 칸

을 이동하라는 명령은 '↑'로 추상화하여 나타낼 수 있고, 전구에 불을 켜라는 명령은 '전구표시'를 사용하여 나타낼 수 있다. 앞으로 두 칸을 이동한 후에 전구에 불을 켜라는 문제의 조건은 위의 기호를 순차적으로 사용하여 '↑', '↑', '전구표시' 로 나타내면 된다. 학생들은 이와 같은 문제를 해결하며, 추상화와 순차적 문제 해결법을 적용하며 문제를 해결하는 경험을 하게 된다.



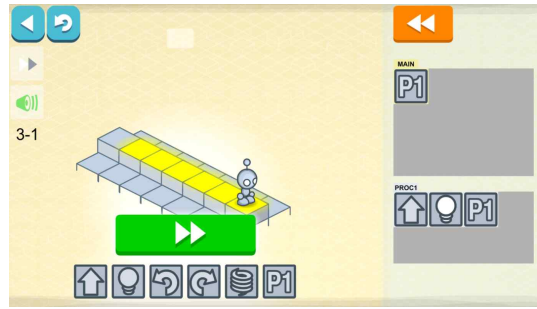

(Fig. 2) Lightbot hour Level 1
for sequential order and abstraction

다음에는 라이트봇 2단계 게임 1-6이다. 2단계에서는 사용할 수 있는 코드의 개수에 제약이 생긴다. (Fig. 3)을 보면, 사용해야 하는 코드는 14개인데, 코드를 입력할 수 있는 MAIN 영역의 칸의 개수는 12개뿐인 것을 확인할 수 있다. 반복되는 패턴을 단축키 PROC1(P1)으로 압축하여 나타낼 수 있어야 해결이 가능한 문제이다. 학생들은 이와 같은 문제를 해결하며, 큰 문제를 작은 문제로 나누어 생각하는 능력, 반복되는 부분을 인식하는 능력을 발전시킬 수 있다.



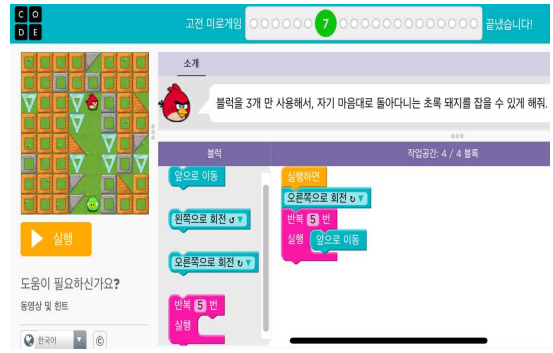

(Fig. 3) Lightbot hour Level 2
for pattern recognition

이어서 3단계의 게임 1-6을 진행한다. 3단계에서는 똑같은 패턴이 계속 반복되는 움직임을 단순한 코드로 표현하는 문제들이 제시된다. (Fig. 4)를 보면, 모두 12개의 코드로 설명 가능한 움직임을, 자동화 과정을 통해 단 하나의 코드 P1으로 압축하여 나타낸 모습을 확인할 수 있다. 이와 같은 루프 구조의 문제 해결 방법을 고민하며, 학생들은 자동화에 익숙해지게 된다.




(Fig. 4) Lightbot hour Level 3
for automation

마지막은 code.org에서 제공하는 고전 미로 게임(https://studio.code.org/hoc/1)이다. 고전 미로게임은 3개~5개 사이의 한정된 수의 블록만을 사용하는 프로그래밍 언어 교육 게임으로, 라이트봇 게임에서 배운 순차, 반복, 순환 등의 개념이 등장한다. 학생들은 고전 미로 게임을 하며 라이트봇 게임에서 배운 문제 해결의 개념들을 블록형 프로그래밍 언어에 적용해볼 수 있다.




(Fig. 5) Classic maze using
only a limited number of blocks

### 3.4. 교육 효과 분석 방법

수업 이전에 46명의 학생들에게 사전검사를 실시하고, 4차시 수업 이후 동일한 학생들을 대상으로 사후검사를 실시했으며, 두 번의 검사 결과를 비교하여 게임에 기반한 프로그래밍 언어 교육 활동의 효과를 분석하였다. 동일한 표본을 대상으로 두 차례 데이터를 수집하여 효과를 분석한다는 평가 방법의 특성을 고려하여 대응 표본 t-test를 통해 효과를 분석하였다.

## 4. 연구결과

### 4.1. 결과 분석 개요

연구 결과를 분석하기 위하여 연구에 참여한 학생 모두를 대상으로 비버 챌린지 문항을 활용하여 설문을 실시하였다. 유효하지 않은 응답이 표기된 설문지는 제외하고, 유효한 설문 응답에 대해서만 결과를 분석하였다.

게임에 기반한 프로그래밍 언어 교육의 효과에 대한 대응표본 t-test를 실시한 결과, 설문지를 통해 향상도를 측정한 모든 부분에서 통계적으로 유의미한 차이가 나타났다.

### 4.2. 측정 영역 -추상화, 자동화, 문제 해결 능력

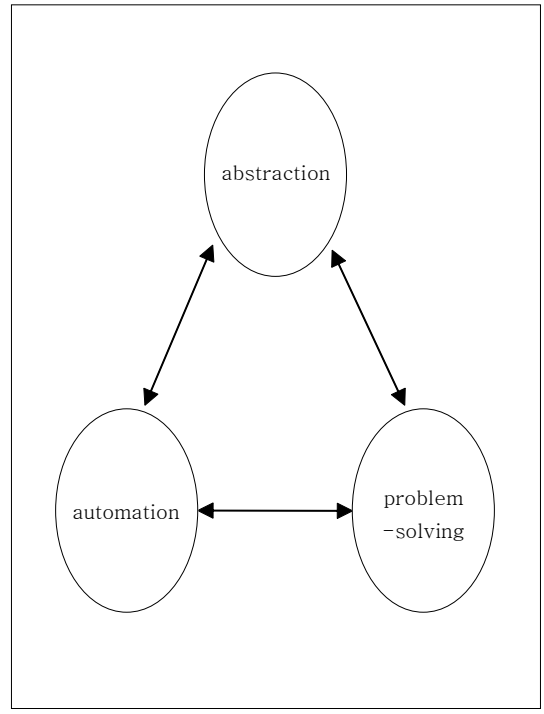
컴퓨팅 사고력의 핵심 요소에 대해 Wing(2008)이나 Yadav(2016) 등은 다음과 같은 견해를 밝히고 있다.

*컴퓨팅 사고력의 핵심은 추상화 능력이다. 특히, 올바른 추상화가 본질이 된다. 추상화 과정에서 어떤 특성을 포함할지, 또 어떤 다른 특성은 제외할지를 판단하는 능력은 컴퓨팅 사고력의 기본이다.*

*우리의 사고력이 가진 힘은 기계적 요소의 도움을 통해 극대화될 수 있다. 컴퓨팅이란 우리가 추상화*
*한 것을 자동화하는 과정이다. 추상화된 요소와 그 요소들 사이의 관계를 기계화하는 자동화 과정을 통해 컴퓨터가 작동되게 된다.*

*컴퓨팅 사고력이 문제 해결 능력 향상에 초점을 맞추고 있고, 특히 학생들이 문제 풀이 과정을 자동화하도록 만들고 있음을 고려하면, 초·중등 교육 관계자들은 컴퓨팅 사고력의 개념을 교육과정에 포함하기 위한 방안을 마련하는 데 많은 노력을 경주해야 한다.*

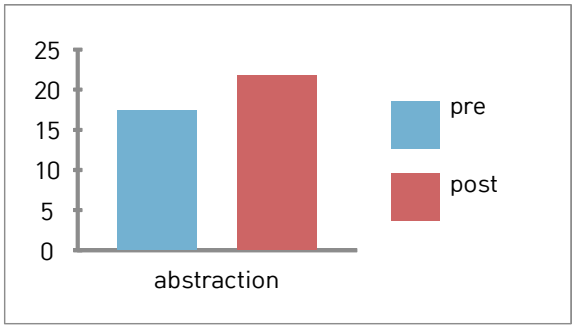
위의 견해를 바탕으로, 컴퓨팅 사고력(Computational Thinking)의 핵심 구성요소를 (Fig. 6)과 같이 정리하여 나타낼 수 있다.




(Fig. 6) Key elements of Computational Thinking and their relations

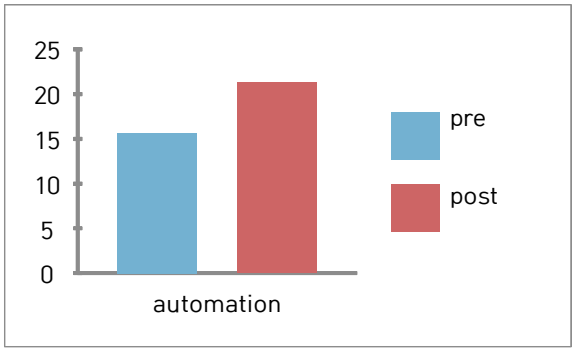
(Fig. 6)에 언급된 핵심 요소의 향상 여부는 교육 활동에 따른 컴퓨팅 사고력의 개발 정도를 평가할 수 있는 척도가 된다. 학생들의 컴퓨팅 사고력 개발 정도를 측정하기 위해 추상화와 자동화, 그리고 문제 해결 능력과 관련된 8개의 문항을 준비하였다. 추상화와 자동화 능력을 확인할 수 있는 문제는 각각 3문항이었으며, 나머지 문제 2개는 문제 해결 능력을 확인할 수 있는 문제로 구성하였다.

### 4.3. 검증 결과



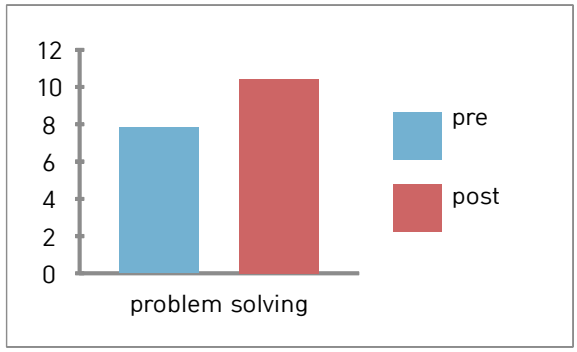

(Fig. 7) Abstraction test result comparison

(Fig. 7)는 추상화 영역에 관한 결과를 그래프로 나타낸 것으로서 p값은 o.oo5로 나타났다. 즉, p<0.05로 통계적으로 유의미한 차이를 보였다. 두 번의 검사 사이에 평균 점수가 4.35점 상승한 것을 확인할 수 있었다. 사전 검사의 평균은 17.39점, 사후 검사의 평균은 21.74점으로 나타났다.




(Fig. 8) Automation test result comparison

(Fig. 8)은 자동화 영역의 결과를 그래프로 나타낸 것으로 p값은 0.004였다. 역시 p<0.05 이므로 통계적으로 유의미한 차이가 나타났음을 확인할 수 있었다. 자동화 영역에서는 추상화 영역보다 평균 점수의 상승 폭이 더욱 크게 나타났다. 사전 검사에서의 평균 점수는 추상화 영역에서보다 낮은 15.65점이었는데, 사후 검사에서는 5.65점이 상승하여 21.30점이라는 높은 점수를 보였다.




(Fig. 9) Problem solving test result comparison

(Fig. 9)에 제시된 문제 해결력에 관한 평가 결과에서도 사전 검사와 사후 검사 사이에서 유의미한 차이가 나타났다. p값은 0.03으로 0.05보다 작아 통계적으로 유의미함을 확인할 수 있었다. 사전 검사는 평균 7.82점. 사후 검사에서는 2.61점이 상승한 평균 10.43점이었다.

(Fig. 7)~(Fig. 9)을 통해 컴퓨팅 사고력의 핵심 구성요소인 추상화, 자동화, 문제 해결 능력이 교육 활동 이후에 유의미한 정도로 향상되었음을 확인할 수 있었다. 이를 통해, 제시된 교수학습 활동이 컴퓨팅 사고력 개발에 효과를 가진다는 것을 연쇄적으로 확인할 수 있다.

다음의 <Table 2>는 t-test 결과를 정리한 표이다. 동일한 46명의 학생들을 대상으로 사전 검사 및 사후 검사가 실시되었으며, 검사 결과에 대하여 평균, 표준편차, t값과 p값 등을 함께 제시하였다.

<Table 2> t-test Result

| factor | test | avg | sd | t | p |
|---|---|---|---|---|---|
| abstraction | pre | 17.39 | 7.58 | 2.07** | .005 |
| | post | 21.74 | 7.77 | | |
| automation | pre | 15.65 | 5.89 | 2.07** | .0004 |
| | post | 21.30 | 6.25 | | |
| problem solving | pre | 7.82 | 5.18 | 2.07* | .03 |
| | post | 10.43 | 5.62 | | |

*$p$<.05, **$p$<.01

## 5. 결론

스크래치 등의 블록형 프로그래밍 언어가 누구나 쉽게 배울 수 있는 프로그래밍 언어 교육 방법이라는 Resnick(2009)의 견해와 달리, 블록형 프로그래밍 언어 수업을 힘들어하는 6학년 학생들과 이런 학생들을 지도하는 데 어려움을 겪는 교사들을 보며 이번 연구를 시작하게 되었다. 블록형 프로그래밍 언어 교육과 관련된 선행 연구를 분석한 결과, 컴퓨팅 사고력을 토대로 프로그래밍 언어 교육이 진행되어야 하며, 컴퓨팅 사고력의 핵심 구성 요소는 추상화, 자동화, 그리고 문제 해결 능력이라는 결론을 도출했다. 그런데, 예비 교원을 위한 교육 프로그램 중에는 컴퓨팅 사고력에 초점을 두고 있는 것이 거의 없어, 교사들이 컴퓨팅 사고력을 초·중등 교육과정에 녹여내는 데에는 어려움이 따른다는 Yadav(2017)의 견해도 확인할 수 있었다.

본 연구에서는 게임을 활용한 프로그래밍 언어 교육을 통해 초등학생들의 컴퓨팅 사고력을 개발하는 수업을 설계하였다. 연구에 앞서 실시된 설문 결과 대다수 학생이 컴퓨터에 흥미를 갖고 있었다. 초등학생의 인지 발달 단계를 고려할 때, 구체적 조작 활동을 활용하면 추상적인 내용 학습에 어려움을 느끼는 학습자의 심리적 부담을 줄일 수 있다는 Piaget(1964)의 견해를 토대로, 컴퓨터 게임이라는 구체적 조작 활동을 활용하는 프로그래밍 언어 교육과정을 설계했다. 프로그래밍 언어 교육 이전에 필요한 추상화, 자동화, 문제해결 능력은 3차시에 걸쳐 라이트봇 게임을 하며 습득하고, 이 과정에서 배운 개념을 마지막 차시에서 고전 미로 게임이라는 기초 단계의 블록형 프로그래밍 언어에서 복습해볼 수 있도록 교수 학습 활동을 계획했다.

본 연구에서 서울 G 초등학교 6학년 2개 학급 학생들에게 위의 4차시 수업을 진행하였고, 수업 전과 후에 비버챌린지 문제를 활용한 사전 검사와 사후 검사가 실시되었다. 검사 결과를 분석하여 도출된 내용으로 게임에 기반한 프로그래밍 언어 교육이 추상화, 자동화, 문제 해결 능력 향상에 유의미한 효과를 가짐을 확인할 수 있었고, 컴퓨팅 사고력의 핵심 구성 요소들의 신장이라는 결과를 통해, 게임 기반의 프로그래밍 언어 교육이 컴퓨팅 사고력 개발에 가지는 효과도 확인할 수 있었다.

본 연구를 통해 초등학생을 위한 블록형 프로그래밍 언어 교육 프로그램을 개발하고, 그 효과성을 제시하고자 하였다. 본 연구에서 제시된 교육과정 설계가 2022 개정 교육과정에서 도입될 인공지능교육과 AI교육이 목표로 하는 컴퓨팅 사고력을 갖춘 인재 양성에 기여할 수 있기를 기대한다.

## 참고문헌


[1] Aho(2012). Computation and computational thinking. *Computer Journal, 55*, 832-835.

[2] Brennan (2012). New framework for studying and assessing the development of computational thinking. *Proceedings of the 2012 annual meeting of the American Educational Research Association*, Vancouver, Canada

[3] Grover.(2013). Computational Thinking in K-12: A Review of the State of the field. *Educational Researcher, 42*(1), 38-43.

[4] Guzdial(2008). Paving the way for computational thinking. Communications of ACM, 51(8), 25-27.

[5] Guzdial(2019). Block-based Programming in Computer Science Education. *Communications of ACM, 62*(8), 22-25.

[6] Jeon, Yongju.(2018). An Exploratory Study on Teaching & Learning and Evaluation Methods using Beaver Challenge in Software Education. *The Journal of Korean Association of Computer Education, 21*(6), 63-82.

[7] Papert(1980). Mindstorms: Children, Computers and Powerful Ideas. New York : Basic Books.

[8] Piaget(1964). Part 1 : Cognitive development in children: Piaget development and learning. *Journal of Research in Science Learning, 2*, 176-186.

[9] Resnick(2009). Scratch: Programming for all. *Communications of ACM, 52*(11), 60-67.

[10] Shin, S.(2018). The Concept of Computational Thinking through Analysis of Computer Education Framework in the United States and its Implications for the Curriculum of Software Education. *Journal of The Korean Association of Information Education, 23*(6), 639-653.

[11] Shin, S.(2019). Designing the Instructional Framework and Cognitive Learning Environment for Artificial Intelligence Education through Computational Thinking. *Journal of The Korean Association of Information Education, 22*(2), 251-262.

[12] Toffler, Alvin(2001). Beyond the Crisis : Korea in the 21st Century.

[13] Weintrop.(2017). Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms. *ACM Transactions on Computing Education, 18*(1), Article 3.

[14] Weintrop.(2019). Block-based Programming in Computer Science Education. *Communications of the ACM, 62*(8), 22-25.

[15] Wing, J. M.(2008). Computational thinking and thinking about computing. *Phil. Trans. R. Soc. A, 366*, 3717-3725.

[16] Yadav(2016). Computational Thinking for All: Pedagogical Approaches to Embedding 21st Century Problem Solving in K-12 Classrooms. *Association for Educational Communications and Technology, 60*, 565-568

[17] Yadav(2017). Computational Thinking for Teacher Education. *Communications of ACM, 60*(4), 55-62.

## 저자소개

**전 병 우**

2016 서울교육대학교 초등교육과 (학사)

2021～서울교육대학교 교육대학원 컴퓨터교육전공 석사과정

2018～현재 서울길원초등학교 교사

관심분야: Computational Thinking, 인공지능교육, 데이터 분석

e-mail: brantbing@gmail.com

**신 승 기**

2017 University of Georgia (Ph.D.)

2016～2017 미국 칼빈슨 정부연구소 연구원

2019～2020 애리조나주립대학교 컴퓨터교육전공 교수

2020～현재 서울교육대학교 컴퓨터교육과 교수

관심분야: Computational Thinking, 인공지능교육, 보편적정보교육

e-mail: skshin@snue.ac.kr