



연구논문/작품 최종보고서

2023학년도 제1학기

Unity 엔진을 활용한 기초코딩 학습게임 개발

안서현 (2017313987)

2023년 4월 19일

지도교수: 이 은 석 서명_____

계획(10)	주제(20)	개념(20)	상세(30)	보고서(20)	총점(100)

<요약>

코로나바이러스 감염증(COVID-19, 이하 ‘코로나19’)의 세계적 유행에 따라 학생들의 학업 수준이 하락한 것으로 나타나면서 학생들에게 충분한 문해력이나 사회성이 제때 갖추어지지 못할 것이라는 우려가 제기되어 왔다. 이 연구에서는, 학생들의 지속적인 학습 의욕과 흥미를 유발할 수 있는 코딩 학습 보조 프로그램을 연구해보고자 하였다.

이 프로그램은 게임의 형태로 기획하여 학생들이 기초수학 및 코딩 학습의 중요성을 친숙하게 체감할 수 있는 환경을 제공하는 것을 목표로 한다. 또한, 연구자 본인의 시스템 설계 경험과 능력을 키우는 데에도 긍정적인 영향을 주고자 한다.

코딩 학습에 관한 몇 가지 연구 선례를 통해, 게임화(Gamification)로서 충분한 게임성을 갖추어 학습자의 흥미를 유발하고, 변수 구조나 알고리즘의 이해뿐만 아니라 근본적으로 문제 해결 능력과 추론 능력에 도움이 될 수 있는 학습 애플리케이션을 개발하는 것을 구체적인 목표로 잡을 수 있었다. 또한, 플립러닝 모델에서 볼 수 있는 사전학습의 효과를 개발하고자 하는 애플리케이션에도 적용해보고, 학습자가 학습 도중에 겪을 어려움이나 오류 상황에 도움을 줄 수 있는 피드백 시스템과 알고리즘을 설계하는 것도 적극적으로 고려하고자 하였다.

코딩 교육 프로그램은 Unity 엔진을 활용해 개발하였고, 주로 10대 학생들로 대표될 수 있는 코딩 초심자 수준에 알맞게 블록 코딩 시스템을 구성하였다. 또한, 해당 블록 코딩 시스템을 기반으로 스테이지 방식의 게임 애플리케이션을 제작하였다. 본 보고서에서는 그 구상 과정과 구현 과정에 관해서 다룬다. 아울러, 내부적으로 구현하는 코딩 시스템의 구성요소를 비롯해 코딩 피드백 시스템을 적용하기 위한 부가 기능 등을 기술하고, 본 연구의 기대효과와 개선 방향을 제시한다.

차례

1. 서론	3
2. 관련 연구	6
3. 제안 작품 소개	10
4. 구현 및 결과	16
5. 결론 및 소감	25
6. 참고문헌	27

1. 서론

가. 작품 제안 배경

코로나바이러스 감염증(COVID-19, 이하 ‘코로나19’)의 세계적 유행으로 인해 사람들의 일상에 큰 변화가 생겼다. 특히, 2020년에는 코로나19의 확산 방지를 위해 대학교뿐만 아니라 초·중·고교에서도 오프라인 수업을 중단하고 온라인 수업으로 전환하는 방역 정책이 세워져 많은 학생이 혼선을 겪기도 했다.

온라인 수업의 학습효과가 오프라인 수업과 비교했을 때 상대적으로 좋지 않다는 점은 꾸준히 지적되어 온 문제다. 특히, 초등학생들의 학업 수준은 코로나19 유행 전과 비교했을 때 국어(읽기·쓰기·말하기)와 수학에서 7~10% 정도 하락한 것으로 분석되고, 온라인 수업과 오프라인 수업에서 나타나는 학습효과의 격차가 초등학교 저학년일수록 크게 나타났다.¹⁾²⁾ 이 때문에 학생들에게 필요한 학습이 제때 충분히 이루어지지 못했기 때문에 학생들의 향후 학업 수행을 위한 충분한 문해력이나 사회성이 부족할 것이라는 우려가 끊임없이 제기되어 왔다.

본 연구의 초기 제안에서는 초등학생의 수학 학습 성취도에 긍정적인 영향을 미칠 수 있는 수학 학습 보조 프로그램을 연구해보고자 했다. 다른 동아시아 주요 국가에 비해 우리나라는 상대적으로 수학 과목에 대한 초등학생들의 학습 흥미가 낮게 나타나는 편이고,³⁾ 이런 조건에서 코로나19의 범유행에 따라 문제 해결을 위한 문해력이 저하된다는 점과 지속적인 문제 해결 경험이 부족하다는 점은 수학 과목의 성취도에 부정적인 영향을 미쳤을 것이기 때문이다.

또한, 본 연구작품에서 연구하고자 하는 수학 학습 보조 프로그램은 게임의 형태로 기획하고자 하였다. 놀이의 요소를 접목한 학습의 기대효과는 학습자의 자발적인 학습 의욕, 다양한 상황에서의 문제 해결 능력 등 여러 측면에서 긍정적으로 나타난다.⁴⁾ 특히, 초등학교 수준의 수학은, 학생들이 배운 내용을 우리의 다양한 일상생활에 알맞게 적용하여 사회성을 기르는 데에도 중요한 역할을 하는 과목이다. 따라서, 수학

1) 임수현, 정은선 "코로나19 기점으로 나타난 초등학교 4, 6학년 수학 학업성취도 변화 분석" 한국초등교육 32.3 pp.249-266 (2021) : 249.

2) 한성민. (2022). COVID-19가 학습 및 생활에 미친 영향: 초등학생 중심으로. 응용경제, 24(2), 55-85.

3) 최지선, 상경아. (2019). 초등학생 수학 성취도에 영향을 미치는 교육맥락변인에 대한 동아시아 5개국 비교. 초등수학교육, 22(3), 167-180.

4) 백영균, 정용석. (2004). 게임기반학습에서 학습자의 게임능력 및 학습능력이 논리적사고력에 미치는 효과. 교육정보미디어연구, 10(4), 119-140.

학습 보조 프로그램의 게임화(Gamification)를 통해 학생들이 기초수학 학습의 중요성을 체감할 수 있는 환경을 제공해보고자 하는 것이 본 연구의 초기 제안 방향이었다.

그러나, 수학 학습 보조 프로그램에 관해서는 이미 다양한 접근이 있고, 흥미를 유발하는 데에는 한계가 있을 것이라는 우려가 있었다. 따라서, 학습 보조 게임이라는 주제는 유지하되, 이 게임이 다루는 영역을 ‘초등학교 수준의 기초수학’이 아닌 ‘중고등학교 수준에서

접근할 수 있는 기초코딩’으로 변경하게 되었다. 대상 연령층이 학업성취도 문제를 가장 많이 겪는 연령층에서 다소 멀어지게 되는 아쉬움은 있으나, 코딩 교육을 통해서도 추론 능력과 문제 해결 능력을 향상하는 데에는 충분한 기대를 모을 수 있을 것으로 생각했다.

나. 작품 설계 목표

본 연구작품의 설계 목표는 첫째로, 게임화를 통해 학생들이 추론 능력과 문제 해결 능력을 향상하기 위한 코딩 학습의 중요성을 체감할 수 있는 애플리케이션을 제공하는 것이다. 코딩은 단순히 특정 언어를 학습하는 영역만을 이야기하는 것이 아니기 때문에, 문제 해결 능력에 도움이 되는 시스템을 설계하는 것이 특히 중요하다. 그러면서도, 대상 학습 연령층을 중고등학교 정도 수준으로 잡아 학습자들이 이후 C나 Python 등의 언어를 본격적으로 학습하는 토대를 마련하고자 한다.

또한, 본 연구작품은 연구자 본인에게도 도전적인 연구과제가 될 것이다. 연구자 본인은 코로나19의 장기화로 학업 수행이 어렵다고 판단해 지난 2021년 한 해 동안 휴학하면서 수학 학원 보조교사로 근무했는데, 이 시기에 여러 학생을 지도하면서 학습 보조 프로그램의 중요성과 필요성을 크게 느꼈다.

문제 풀이에 학생들이 어려움을 겪는 유형이 꽤 다양하다는 것을 볼 수 있었다. 어떤 단원에서 다루는 연산 방법을 익혔더라도 그 원리에 관한 이해가 부족해 특정 유형에서 일관된 계산 오류를 범하는 경우, 문제에서 요구되는 풀이가 왜 필요한지 이해하지 못한 상태로 맹목적으로 풀이 방식을 유형화하는 경우, 문장의 호응 구조에 관한 이해가 부족해 문제 해석에서부터 어려움을 겪는 경우 등이었다.

당시 이런 이유로 학습에 어려움을 겪던 학생들의 이해를 돕기 위해 가장 많이 쓴 방법은 학생들에게 친숙한 소재로 문제 상황과 비슷한 예를 들어 관심을 유도하는 것이었다. 연산 영역에서는 학생들이 오개념이 잡히지 않은 상태로 꾸준히 연습할 수

있도록, 학생들이 실수한 부분이 있다면 어떤 부분에서 왜 실수한 것인지 꼼꼼히 살펴봐 주기도 했다. 이처럼 학습을 보충하는 과정이 게임 프로그램을 통해 이루어진다면 학생들의 학습 흥미를 유발하는 데에 큰 도움이 될 것으로 생각했다.

따라서, 본 연구에서 궁극적으로 추구하고자 하는 목표는 다음과 같이 정리할 수 있다. 첫째로, 문제 해결 능력에 실질적으로 도움이 될 수 있는 코딩 교육 프로그램을 설계하되, 대상자가 학습하는 과정에서 적절한 피드백이 이루어질 수 있는 구조를 갖추는 것이다. 또한, 본 연구를 통해 연구자 본인의 시스템 설계 경험과 능력을 키우는 데에도 긍정적인 영향을 주는 것도 또 다른 목표가 될 것이다.

다. 작품 개요

본 연구에서 제안하는 작품은 퍼즐을 기반으로 한 스테이지 형식의 블록 코딩 게임 애플리케이션이다. 일반적으로 우리가 작성하는 텍스트 코드의 흐름이 가지는 특성을 생각해봤을 때 이 부분을 시각화하기 위해 ‘길을 따라 필요한 요소를 적재적소에 배치하는 퍼즐’의 형식을 구상할 수 있었고, 점점 더 도전적인 과제로 나아가기 위해 ‘스테이지’ 형식의 게임 구조를 구상할 수 있었다.

제안한 작품에서는 코딩에 사용하는 요소를 ‘블록’으로 구현하고, 이 블록의 유형을 각각 변수 블록, 기호 블록, 명령문 블록으로 구분하였다. 먼저, 변수 블록을 통해 기초적인 변수의 유형을 학습할 수 있도록 하고 기본적인 정수형과 실수형 변수의 연산 차이점 등 각 변수의 특징을 이해할 수 있도록 하였다. 또한, 변수 연산 결과가 어떤 변수형으로 표현이 가능한지를 기호 블록으로써 학습할 수 있게 하고, 필요한 명령을 알맞은 명령문 블록으로 조합함으로써 코딩을 본격적으로 경험해볼 수 있게 하였다.

스테이지를 제시하여 적절한 커리큘럼을 따라 순차적인 학습이 가능하게 하면서도, 작성한 코드를 프로그램 내에서 자체적으로 텍스트로 변환해 정답 텍스트와 비교해보며 피드백을 줄 수 있는 구조를 갖추하고자 하였다. 학습자가 겪을 어려움이나 오류에 대비하여 syntax error 피드백, runtime error 피드백, 오답 피드백 등을 제시하고자 하였다.

2. 관련 연구

코딩 교육을 위한 프로그램의 대표적인 예시로 미국의 MOT Media Lab의 ‘스크래치(Scratch)’와 우리나라 엔트리 교육연구소의 ‘엔트리(Entry)’를 들 수 있다. 이들은 텍스트라는 요소를 블록으로 표현한 블록 코딩의 예시이다.

홍익대학교의 한 연구팀은 이들이 학습 초기에는 학습자의 흥미를 유발하는 데에는 효과가 있다고 하면서도, 중급, 고급 단계로 갈수록 이들 프로그램이 실질적인 논리 사고 훈련보다 흥미 위주의 입출력 행위만을 반복시키며 자발적인 자기주도 학습을 유도해내는 데에는 한계를 보이고, 심하게는 알고리즘적 사고에 익숙지 않은 학습자의 학습 동기를 상실시키기도 한다는 점을 지적한다.

이 연구팀은 메이커스랩과 산학협동으로 코딩교육용 게임을 개발하고 초등·중학생 300명을 대상으로 설문을 진행하여 코딩 학습에 대한 몰입과 학습 효과를 증진할 방안을 고찰해보았고, 그 결과로, 게임 내에서 적절한 보상을 제공하고 풍부한 스토리라인을 제공하는 등 학습자의 학습 동기를 꾸준히 유발할 만한 게임성을 갖추는 것이 중요하다는 결론을 제시하였다.⁵⁾

처음으로 코딩 교육을 접하게 되는 유아 및 초등 저학년을 대상으로 한 ‘언플러그드’ 코딩 교육의 커리큘럼에 관한 연구에 따르면, 코딩 교육은 단순히 소프트웨어의 활용 능력만을 키워주는 것이 아니라 생각하는 방법과 문제 해결 능력, 추론과 논리적사고력 등의 더 큰 학습 능력을 갖추기 위한 토대 교육을 포함하는 것이라고 지적한다.

이 연구는 텍스트 코딩과 같은 본격적인 코딩 학습 단계에 들어서기 전 유아 및 초등 저학년 단계에서 선행되어야 할 근본적인 컴퓨팅 사고 학습 과정을 제시하며, 이에 따라 한국의 코딩 교육 교과 과정에서 제시하는 학습자의 연령대별 교육용 언어(방식)를 정리하기도 했다. 언플러그드 코딩을 통한 컴퓨팅 사고와 문제 해결 능력을 키우는 과정이 유아 및 초등 저학년 단계에서 선행된 후, 초등 고학년 단계부터는 블록 코딩을 통한 논리 표현을 학습하는 것이 권장된다.⁶⁾

효과적인 코딩 교육 방식에 관한 다른 한 연구에서는, 기존의 수업 진행 과정을 거꾸로 진행하는 ‘플립러닝(Flipped Learning)’ 모델을 접목하는 것을 제안한다. 플립러

5) N. Kim, "A Study on the effect of coding education and improvement of learning achievement using educational game," Journal of Korea Game Society, vol. 17, no. 4. Korea Academic Society of Games, pp. 161-168, 31-Aug-2017.

6) 안성혜. "유아용 코딩아트 교육을 위한 커리큘럼 설계" 예술교육연구 17, no.2 (2019) : 43-60.

닝은 수업 전에 여러 매체를 통해 강의 내용을 사전 숙지한 후에 수업에서는 주어진 과제를 해결함으로써 강의식 수업의 수동적 학습자가 능동적인 수업의 주체가 되게 한다.

이 연구에서 제안된 학습 모델에서는 사전학습 단계에서 특정 문법을 학습하기 전에 결과물에 관한 다양한 예시를 제공하여 학습자가 이를 통해 해당 문법의 활용과 응용 여지에 대해 고민해볼 수 있도록 한다. 또한, 이와 같은 학습자 중심의 수업 디자인을 적용하여 실제 학습을 진행해본 결과, 긍정적인 학습 효과가 있었다는 결론을 제시한다.⁷⁾

해외의 연구에서도 코딩 학습에 흥미를 느끼게 하는 것이 중요하다고 이야기한다. 한 연구에서는 경험된 흥미가 학생들의 코딩 학습에 주는 직접적이고 긍정적인 영향, 경험된 흥미가 학생들이 코딩을 대하는 태도에 주는 직접적이고 긍정적인 영향, 학생들이 코딩을 대하는 태도가 코딩 학습에 주는 직접적이고 긍정적인 영향, 경험된 흥미가 학생들의 코딩 학습에 주는 간접적이지만 긍정적인 영향이라는 네 가지 척도를 통해 흥미의 중요성을 연구하기도 했다.

이 연구에 의하면, 놀이와 학습의 연계를 통해 학습자들이 긍정적인 태도를 보인다는 데에서는 학계에서도 다소 대립적(controversial)인 의견이 있기도 하고, 학습 과정에서 '재미'라는 요소가 자리하기에는 다소 시기상조라는 연구가 있었다고 한다. 그러나 코딩 학습에 관해서는 학습 과정에서 재미를 느끼게 하는 것이 유의미한 결과가 있었다는 연구 결과를 구체적인 수치화 관계를 제시하며 끌어냈고, 이 부분으로 코딩 학습에 재미 요소를 추가하는 시도가 중요함을 언급한다.⁸⁾

해외의 다른 한 연구에서는 객체 지향 프로그래밍(Object-oriented Programming)을 고등학교 학생들에게 가르치기 위해 CodeCombat이라는 플랫폼을 활용해 코딩 학습을 진행하기도 했다.

이 연구에서는, 기본적으로 게임화(Gamification)는 커리큘럼 목표에 부합해야 하고 게임화 구조나 디자인은 과목 주제와 밀접히 연관되어야 한다고 이야기하며, 즉각적인 지도와 피드백의 반복을 통해 궁극적으로 게임 진행 양상에 따른 학습 프로세스를 이해하는 효과를 꾀해야 한다고 제안한다. 또한, 컴퓨터 과학 영역의 40~50% 정도에

7) S.-J. Kim and D.-E. Cho, "A Study on Learning Model for Effective Coding Education," Journal of the Korea Convergence Society, vol. 9, no. 2, pp. 7-12, Feb. 2018.

8) G. Tisza and P. Markopoulos, "Understanding the role of fun in learning to code," International Journal of Child-Computer Interaction, vol. 28. Elsevier BV, p. 100270, Jun-2021.

서는 학교 수준 혹은 대학 수준의 학생들이 학습을 지속하지 못하고 다른 학습 영역으로 이탈하는데, 전 세계 학생 중 3분의 1 정도가 입문 수준의 컴퓨터 과학 교과를 통과하지 못하는 정도의 영역이라는 문제를 지적한다. 이를 개선하기 위해 여러 교육자와 연구자가 컴퓨터 교과 학습을 더 흥미롭고 재밌게 할 방안을 꾸준히 연구하고 있다고 하며, 게임화도 그 방안 중 하나로 제시되고 있다.

이 연구의 CodeCombat 학습 모델은, 객체 지향 프로그래밍과 같이 컴퓨터 과학에서 특히 중요한 커리큘럼을 적절히 분배하면서도 이를 효과적으로 학습시키기 위해 게임 모델을 도입함으로써 학생들이 게임에 필요한 요소를 코딩으로 직접 완성하게 하는 뚜렷한 목표를 설정해주었다.⁹⁾

더 효과적인 코딩 교육 효과를 위해 코드 피드백 알고리즘을 제안한 연구도 있었는데, 이 연구에서는 크게 리팩토링(Refactoring), 구조 정렬(Structure Alignment), 블록 교정(Block Repair)의 세 단계로 코드 피드백을 진행할 것을 제안한다.

첫째로 리팩토링은 정해진 컨트롤플로우(control flow)에 맞도록 코드 구조를 교정하는 것이다. 이를테면 else 문이 정의되지 않은 if 문 코드에 빈 else 문을 삽입하여 일관된 코드 구조를 정립하는 것이다.

이어서 둘째로, 구조 정렬은 비슷한 기능을 하는 두 비교 대상 코드 일부분을 대응시키는 것이다. 그렇게 해서 일치하지 않는 부분이 있다면 그 부분에 버그가 있는 것으로 생각하고 여기에 구조 변이(structure mutation) 작업을 거쳐 정답 코드와 비슷한 형태를 만드는 것이다.

마지막으로, 블록 교정 과정에서는 변수 매핑 등을 진행한 후 각 블록에서의 입출력에 관한 사양을 생성한다. 그런 뒤에 정답 코드의 블록 상태와 일치시키는 작업을 본격적으로 시행하고, 피드백을 생성한다. 이처럼 단순히 syntax 오류나 runtime 오류만으로는 피드백해주기 어려운 부분에 관해서도 심층적인 코드 검토가 가능해질 것으로 기대할 수 있다.¹⁰⁾

이들과 같은 연구 선례를 통해 본 연구의 방향성을 조금 더 뚜렷하게 잡아볼 수 있

9) O. Karram, "The Role of Computer Games in Teaching Object-Oriented Programming in High Schools - Code Combat as a Game Approach," WSEAS TRANSACTIONS ON ADVANCES in ENGINEERING EDUCATION, vol. 18, World Scientific and Engineering Academy and Society (WSEAS), pp. 37-46, 02-Apr-2021.

10) Y. Hu, U. Z. Ahmed, S. Mechtaev, B. Leong and A. Roychoudhury, "Re-Factoring Based Program Repair Applied to Programming Assignments," 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA, 2019, pp. 388-398.

었다. 게임화를 목적으로 하는 만큼, 충분한 게임성을 갖추어 학습자의 흥미를 유발할 수 있는 시스템적인 구조를 갖추는 것이 특히 중요하다고 생각했다. 또한, 코딩 교육은 단순히 특정한 코딩 언어나 프로그래밍을 학습하는 영역만을 의미하는 것은 아니기 때문에 변수 구조나 알고리즘의 이해뿐만 아니라 근본적으로 문제 해결 능력과 추론 능력에 도움이 될 수 있는 학습 애플리케이션을 개발하는 것이 목표가 되어야 했다.

코딩 학습 보조 프로그램으로서 더 좋은 학습 효과를 얻기 위한 고려 요소도 찾아볼 수 있었다. 플립러닝 모델에서 볼 수 있는 특정 주제에 관한 사전학습의 효과를 본 연구에서 개발하고자 하는 애플리케이션에도 적용할 수 있도록 하고, 학습자가 학습 도중에 겪을 어려움이나 오류 상황에 도움을 줄 수 있는 피드백 시스템과 알고리즘을 설계하는 것도 중요하게 생각하면서 본 연구를 진행하였다.

3. 제안 작품 소개

가. 콘셉트 구상

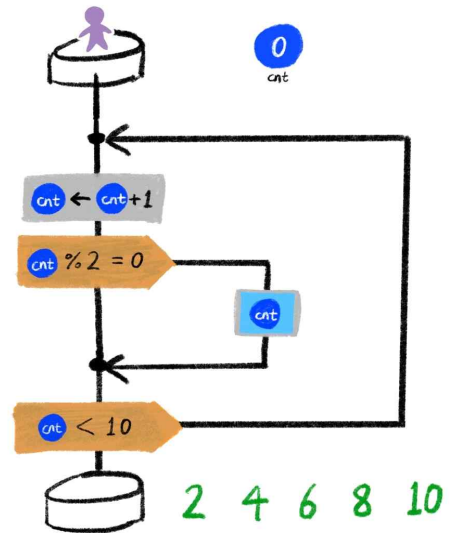
본 연구작품은 코딩 학습 보조 게임 프로그램이다. 코딩에 필요한 기초적인 논리력을 중고등학교 수준에서 친숙하게 학습할 수 있도록 하는 것이 이 프로그램의 목표가 된다. 또한, 이 프로그램을 이용하는 학생들이 이후 블록 코딩이나 C, Python 등으로 본격적인 코딩 학습을 시작할 때 좀 더 쉽게 학습할 수 있는 계기를 마련하는 것이 좋을 것이다.

그러기 위해서 코딩이라는 주제를 어떤 테마에 녹여낼 것인가를 고민할 필요가 있었다. 우리가 일반적으로 생각하는 줄글 코딩은 코딩을 처음 학습하는 이의 입장에서는 다소 복잡하고 막연하게 다가올 수 있다는 점을 고려해보니, 코딩을 학습자에게 친숙한 형태로 시각화하는 것이 중요하다는 걸 알 수 있었다.

일반적으로 우리가 작성하는 코드는 특정한 순서나 규칙으로 명령어들을 나열한 것이고, 마치 길을 따라가는 것처럼 동작한다. 이 부분을 시각화하기 위해, 길을 따라가면서 필요한 요소를 적재적소에 배치하는 ‘퍼즐’의 형식을 구상하였다.

예를 들어 [그림 1]에서는, int 형식을 나타내는 파란 동그라미 모양의 변수 ‘cnt’가 0에서 시작하여 10이 될 때까지 길을 따라 이동하면서 짝수가 될 때마다 그 값을 출력하는 동작을 수행한다. 명령어들은 그림 내에서 node의 형태로 존재하며, 다음 명령어로 향하기 위한 길이 있다. 일반적으로는 아래로 향하는 길이지만, if 문과 같이 특수한 분기 상황에서는 다른 방향으로 가면서 상황에 맞는 명령어를 향해 이동한다.

쉽게 출제하는 영역의 스테이지에서는, [그림 1]과 같이 흠뻑을 구분하는 문제, 특정한 수의 합을 구하는 문제, 수나 문자열의 대소나 우선도를 비교하는 문제 등을 출제할 수 있을 것으로 생각한다. 어려운 주제로 간다면, 실생활과 연계되는 계산 문제, 하노이 탑과 같이 규칙성을 활용하는 문제도 생각해볼 수 있을 것이다.



[그림 1] 출제 예시 구상

나. 블록 코딩 시스템의 구성

앞서 구상한 내용과 같이 [그림 1]과 같은 퍼즐 형태의 블록 코딩을 구성하기 위해, 어떤 블록을 사용할 것인지를 판단하였다. 블록의 유형에 따라 변수 블록, 기호 블록, 명령문 블록으로 구분하기로 하였고, 그 내용은 아래와 같다.

A. 변수 블록












변수 블록에서는 주요 5개 변수형을 다룬다. 학습 대상을 고려하여 실제 다른 언어에서 쓰이는 변수형의 특징과는 다소 차이가 있으며, 구체적인 내용은 [표 1]과 같다.

유형	블록 이미지	설명
int		본 프로그램에서는 ‘정수형’ 변수로 가정하며, 프로그램의 이용 대상층을 고려하여 long 등의 확장형 정수는 고려하지 않는다.
float		본 프로그램에서는 ‘실수형’ 변수로 가정하며, 프로그램의 이용 대상층을 고려하여 double 등의 확장형 실수는 고려하지 않는다.
char		문자 1개에 해당하는 변수이다. ASCII 코드에 따른 int와의 상호호환을 고려한다.
string		문자열에 해당하는 변수이다. 다만, 프로그램의 설계 수준을 고려했을 때 char의 array 형태로 간주하기보다는 1개 이상의 문자로 구성된 독립된 문자열로 간주한다.
bool		True 혹은 False를 판단하는 변수이다.

[표 1] 변수형 블록 유형과 그 개체의 이미지

B. 기호 블록


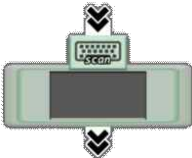
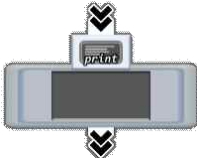
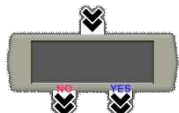
기호 블록은 기능에 따라 대입, 연산, 비교 블록으로 구분되고, 이 블록을 포함하는 식 블록의 역할에 영향을 준다. 예를 들어 “int + int” 형태의 연산을 수행한 결과는 int이므로 “int + int”를 표현하는 식 블록은 int의 역할을 하는데, 반면에 “int < int” 형태의 비교를 수행한 결과는 bool이므로 이를 표현하는 식 블록은 bool의 역할을 하게 되는 셈이다. 구체적인 내용은 [표 2]에 정리되어 있다.

유형		블록 이미지	기능
대입 블록	←		오른쪽의 변수값을 좌측 변수에 대입한다. 이용자의 학습 수준을 고려하여 ‘a = 3’의 형태보다는 ‘a ← 3’의 형태로 표현하는 것을 고려하였다.
연산 블록	+		좌우에 있는 값에 대하여 연산을 수행하며, 이 블록을 포함한 식 블록은 해당 변수 유형에 맞는 값을 반환한다. 특히, 덧셈은 “char + int”와 같은 형태나 “char + string”과 같은 형태도 대응하여 적절한 형 변환이 이루어지도록 한다. 형 변환의 기준은 코딩 언어인 C#의 기준을 따르게 된다. 한편, 덧셈 외의 다른 연산은 int와 float 사이에서의 연산만을 고려한다.
	-		
	×		
	÷		
	%		
비교 블록	=		좌우에 있는 값을 비교하며, 이 블록을 포함한 식 블록은 bool 값을 반환한다.
	>		
	≥		
	<		
	≤		

[표 2] 기호 블록 유형과 그 개체의 이미지

C. 명령문 블록

변수 블록과 기호 블록을 조합하는 곳이면서 프로그램이 따라가는 명령의 틀이 되는 블록이다. 학습 대상이 추후 다른 언어로 텍스트 코딩을 학습하는 데에 도움을 주기 위해 scan, print 등의 입출력, if를 통한 분기까지 구상하였다. 구체적인 내용은 [표 3]과 같다.

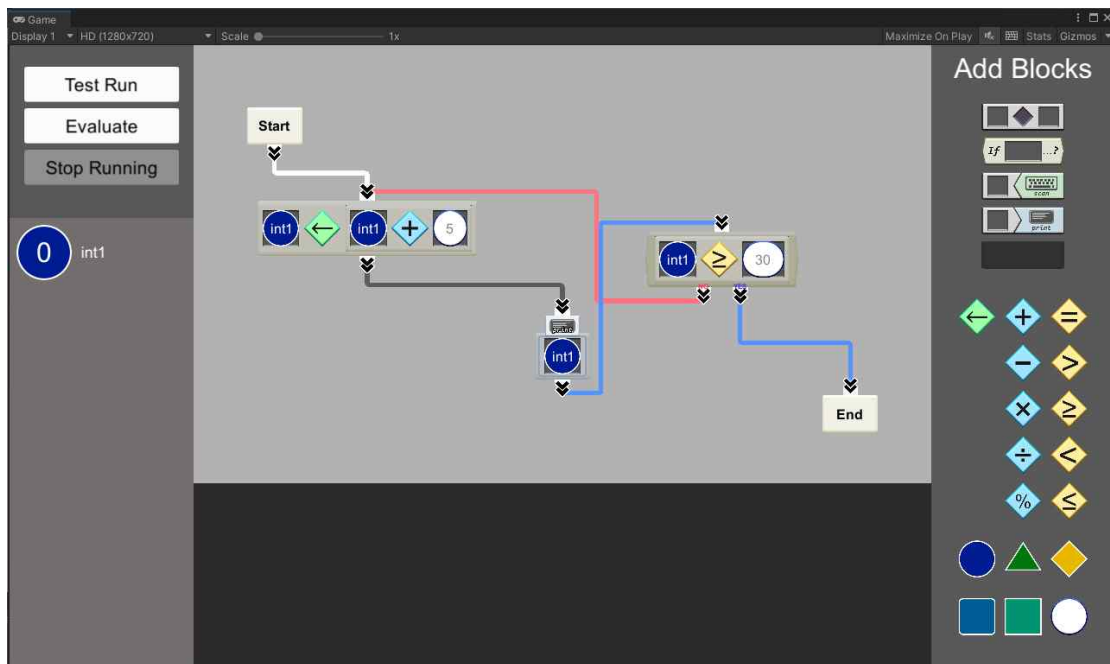
유형	블록 이미지	설명
inst		가장 기본적인 명령문 블록이자 식 블록으로 사용하는 블록이다. “(변수 블록) (기호 블록) (변수 블록)”의 형태로 구성이 되어있고, 변수 블록 대신 식 블록을 추가하여 “a + (b - c)”와 같은 형태의 복합 연산으로 확장할 수도 있다.
scan		사용자로부터 직접 값을 입력받아 변수에 저장하기 위한 블록이다. 블록 내에는 값을 입력받을 변수 블록만 들어갈 수 있다.
print		특정 변수의 값이나 내용을 출력하기 위한 블록이다. 블록 내에는 출력할 수 있는 모든 형태의 블록이 들어갈 수 있다.
if		조건에 따라 분기를 발생시키기 위한 블록이다. 블록 내에는 bool을 반환하는 형태의 블록이 들어갈 수 있다.

[표 3] 명령문 블록 유형과 그 개체의 이미지

또한, 각 명령문 블록에는 화살표 진입선과 진출선이 있다. 이들을 활용해 명령문 블록 간 연결을 할 수 있고, 그렇게 명령문의 동작 순서를 정하게 된다.

다. 코딩 교육 애플리케이션의 구조

구상한 블록 코딩 시스템으로 개발하고자 하는 코딩 교육 애플리케이션은 학습 흥미를 위해 게임의 형태를 따르도록 하고, 게임의 여러 진행 방법 중 스테이지 형식의 진행 방법을 선택했다. 스테이지 방식의 게임은 게임 내의 이야기 진행이나 난이도 변화에 따라 성취도를 직접 확인하기에 유리하다. 특히, 퍼즐 유형의 게임에서 과제나 문제 풀이 요소를 하나씩 주기 위해 이 방식을 적극적으로 활용하고 있는 것을 알 수 있다. 따라서 본 연구에서도 이 방식을 활용하여 코딩 과제를 순차적으로 주는 시스템을 갖추고자 하였다.



[그림 2] 블록 코딩 스테이지의 인터페이스

각 스테이지에서는 사용자에게 학습시키고자 하는 내용으로 코딩 과제를 제시하고, 사용자는 필요한 정답을 출력할 수 있는 형태의 코드를 블록으로 완성하게 된다. 사용자는 변수 블록, 기호 블록 등을 알맞은 명령문 블록에 적절히 삽입하여 어떤 명령을 차례로 내릴지 결정한다.

전체적인 인터페이스는 [그림 2]와 같다. 좌측에는 코드 실행 혹은 채점을 위한 버튼과 변수 상태를 확인할 수 있는 창이 마련되어 있고, 우측에는 명령문 블록, 변수형 블록, 기호 블록 등을 추가할 수 있는 버튼이 있다. 중앙 상단의 공간이 코딩 블록을 배치하는 곳이며, 중앙 하단은 디버깅 및 콘솔 창이 된다. 모든 명령문 블록의 순서가 정해지면, 사용자는 이 프로그램을 동작시킬 수 있고, 각 명령문을 거칠 때마다

다 선언된 변수들에 저장되는 값들이 어떻게 변화하는지 시각적으로 살펴보며 자신이 작성한 코드가 올바르게 동작하는지 확인해볼 수 있다.

또한, 각 스테이지에서는 요구되는 정답이 있으므로, 작성된 블록 코드를 텍스트로 따로 변환하여 모범답안과 비교해볼 수 있도록 피드백 기능을 확장하고자 하였다. 어셈블리 코드만큼은 아니지만 마치 그와 유사한 형태로 변환하여 피드백 과정에서 리팩토링 등을 쉽게 반영하고자 하는 것이다.

4. 구현 및 결과

가. 개발 과정 및 단계 설정

본 프로그램은 Unity 엔진 2020.3.44f1 LTS 버전을 사용하여 개발 중이다. 대상 플랫폼은 PC 환경으로 구상하였다. 다만, 게임의 형태로 개발한다는 점을 고려하여 Android 애플리케이션의 형식으로도 호환하는 것을 나중에 충분히 염두에 둘 수 있는 구조로 설계하고자 했다. 또한, 작성되는 소스 코드의 언어는 주로 C#이다.

원활한 연구 진행을 위해 아래와 같이 애플리케이션을 개발하는 과정을 아래와 같이 단계별로 설정하였다. 주로 기초적인 기능을 먼저 구현한 후 확장하는 형태이다.

(1) 프로그램 개발 기획

- 어떤 콘셉트로 어떤 프로그램을 개발할 것인지 구상한다.
- 이하와 같은 개발 단계를 두어 개발 목표를 설정하였다.

(2) 블록 코딩 프로그램 기초 형태 구현

- int 변수형과 inst 블록, 대입 및 연산 블록만 사용하여 프로그램의 기초 형태를 구현한다.
- 변수 블록이나 기호 블록을 inst 블록의 슬롯에 삽입할 수 있도록 한다.
- 코드 실행 시, 각 명령문 블록에서 하위 슬롯의 정보를 읽어 요구된 동작을 하도록 한다.

(3) 명령문 블록 간 연결선 구현

- 명령문 블록끼리의 동작 순서를 나타내는 연결선 기능을 구현한다.

(4) 변수 블록과 기호 블록 확장 구현

- float, char, string, bool 변수형을 확장 구현한다.
- 비교 연산 블록이 적용되도록 한다.
- inst 블록이 식 블록으로 동작할 때 나타내는 자료형을 판단할 수 있도록 한다.

(5) 오브젝트 prefab화

- 블록 오브젝트를 prefab으로 저장한다.
- 사용자가 필요에 따라 버튼 등을 클릭하여 오브젝트를 추가할 수 있도록 한다.

(6) 블록 코드 텍스트화

- 작성한 블록 코드를 텍스트 형식으로 변환하여 내보낼 수 있도록 한다.

(7) 명령문 블록 확장 구현

- scan 블록과 print 블록을 구현한다.
- 테스트케이스에 따라 정답 여부를 판별하는 시스템을 구현한다.
- if 블록과 for/while 블록을 구현한다.

(8) 스테이지 형식 구현

- 스테이지 이동 기능을 구현하여 게임 애플리케이션으로서의 틀을 갖추어 나간다.
- 각 스테이지에 낼 문제를 결정하여 반영한다.

(9) 코딩 피드백 알고리즘 적용

- 블록 코딩 내용을 변환한 텍스트 정보를 활용하여 사용자에게 코딩에 관해 피드백을 주는 알고리즘과 기능을 적용한다.

(10) 디버깅 및 그래픽 보강


- 개발 과정 전반에 걸쳐 요구 사양에 맞지 않는 오류를 개선한다.
- 이미지 스프라이트를 필요에 따라 교체하여 심미성과 가독성을 개선한다.

나. 오브젝트 요소 기초 구현

프로젝트 파일 내에서 명령문 블록과 변수 블록, 값 블록, 기호 블록은 Prefab으로 관리하여 사용자의 필요에 따라 추가가 되도록 하였다. 또한, 블록을 조립하는 기초가 되는 블록이 특히 명령문 블록 중에서도 inst 블록인데, 해당 오브젝트의 구성은 [표 4]와 같다.

블록 코딩 프로그램에서 명령문 블록은 연결선을 따라 정해진 순서로 동작하게 되고, 한 명령문 블록 안에서는 두 변수 블록 슬롯과 기호 블록 슬롯에 삽입된 블록의 내용에 따라 주어진 연산을 수행한다. 다만, 좌측이나 우측 슬롯에 식 블록이 삽입된 상황이라면, 그 식 블록의 내부에서 먼저 연산을 수행하여 특정 값을 반환한다.

예를 들어, “[$a \leftarrow [[3 \times a] + b]$]”라는 식이 여러 식 블록으로 구현이 됐다면, “ $3 \times a$ ”를 먼저 수행하고 그 결과에 대해 “[$3 \times a] + b$ ”를 수행한 뒤, 이 값을

오브젝트	하위 오브젝트	설명
	좌측 슬롯	<p>식 블록이나 변수 블록, 값 블록을 삽입하는 블록이다. 단, 중앙 슬롯에 대입 기호 블록이 사용되면, 변수 블록만 삽입할 수 있다.</p> <p>시스템상 scan 블록이나 print 블록에서의 유일한 슬롯이 되며, 이 경우에는 변수 블록만 삽입할 수 있다.</p>
	우측 슬롯	<p>식 블록이나 변수 블록, 값 블록을 삽입하는 블록이다.</p> <p>scan 블록이나 print 블록에서는 노출되거나 사용되지 않는다.</p>
	중앙 슬롯	<p>기호 블록을 삽입하는 블록이다.</p> <p>scan 블록이나 print 블록에서는 노출되거나 사용되지 않는다.</p>
	연결선 진입부	<p>이전 명령문 블록에서의 연결선이 이어지는 부분이다.</p> <p>다른 명령문 블록에서 생성된 연결선을 드래그하다가 이곳에 클릭하면 연결이 완료된다.</p> <p>이 블록이 식 블록으로 사용되어 다른 명령문 블록의 슬롯에 삽입되는 경우엔 이 오브젝트는 노출되지 않는다.</p>
	연결선 진출부	<p>다음 명령문 블록으로 연결하기 위한 부분이다.</p> <p>클릭으로 연결선을 생성할 수 있다.</p> <p>이 블록이 식 블록으로 사용되어 다른 명령문 블록의 슬롯에 삽입되는 경우엔 이 오브젝트는 노출되지 않는다.</p> <p>if 블록에서 사용되는 경우엔 같은 오브젝트를 하나 더 두어, 판단 기준이 되는 bool 자료의 값에 따라 진행을 분기할 수 있도록 한다.</p>

[표 4] 명령문 블록 오브젝트의 구성

변수 a에 대입하는 것이다.

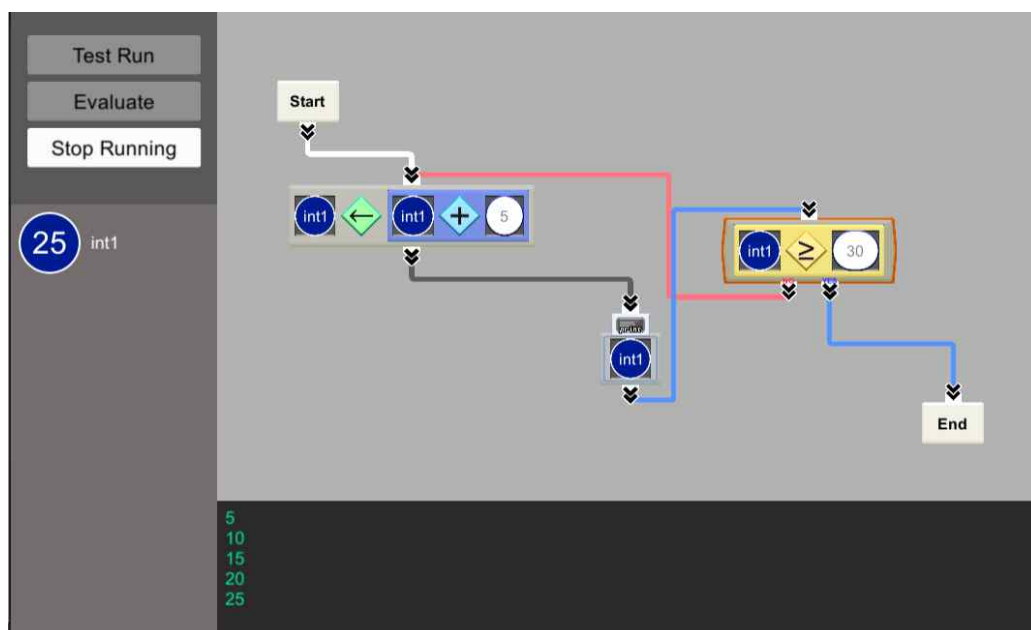
명령문 오브젝트나 변수 오브젝트, 기호 오브젝트 등에는 Collider 컴포넌트를 삽입하였다. 마우스로 오브젝트를 선택해 이동하는 이벤트가 발생하면 이 Collider 컴포넌

트가 활성화되면서 다른 오브젝트 근처에 다다랐는지를 판단하게 된다. 명령문 오브젝트 내에 존재하는 슬롯 오브젝트에도 Collider 컴포넌트를 삽입하여, 특정 범위 내에서의 슬롯과 삽입 객체의 충돌이 감지되면 그 오브젝트를 슬롯에 장착하도록 하였다.

오브젝트가 슬롯에 장착된 이후에는 해당 명령문 블록의 상태를 갱신하도록 하였다. 예를 들어, int 형식의 변수 a를 명령문 블록의 한 슬롯에 장착했다면, 그 즉시 슬롯은 해당 변수 클래스를 참조하게 된다. 이처럼 명령문 블록 내의 모든 슬롯이 변수 혹은 기호 오브젝트를 참조할 수 있게 될 때 비로소 동작이 가능한 상태가 되는 것이다.

다. 실행 프로세스 기초 구현

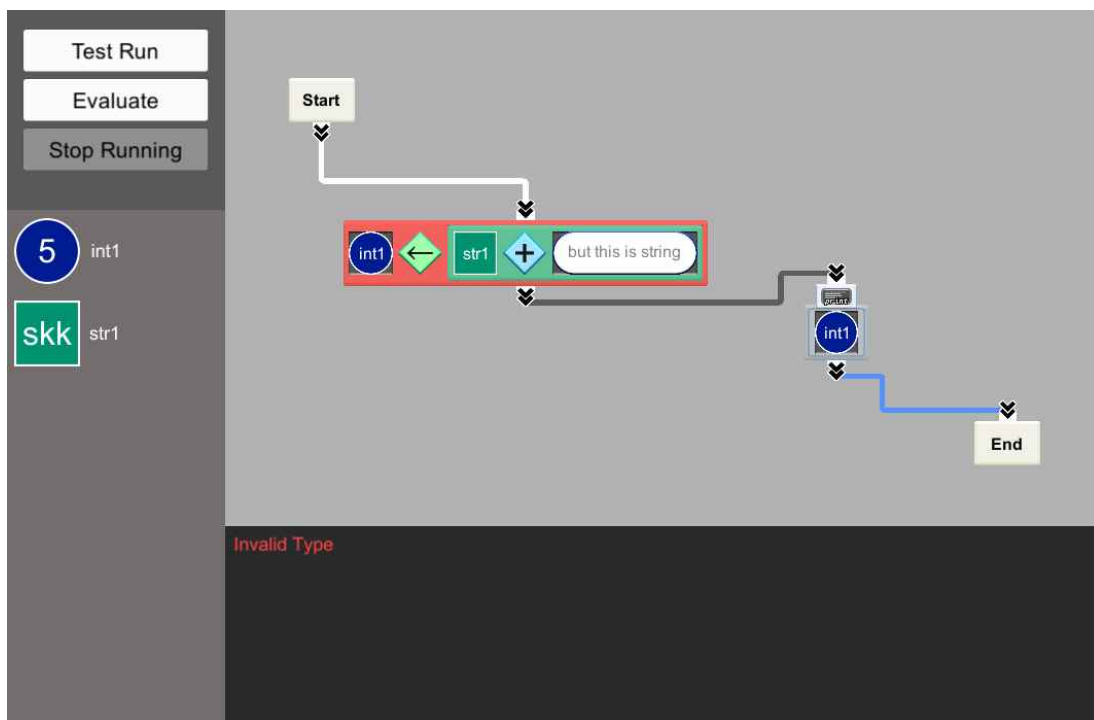
[그림 3]은 [그림 2]에서 작성된 예시 코드를 실제로 실행한 것이다. 순서에 따라 각 명령문은 2초마다 하나씩 실행되도록 설정되었다. [그림 3]의 코드에서는 'int1'에 5씩 더하여가며 int의 값을 출력하고 있다. 'int1'에 저장된 값이 무엇인지를 좌측 공간에서 실시간으로 확인할 수 있고, print 블록을 통해 출력되는 값이 하단 콘솔에 적히는 것을 볼 수 있다.



[그림 3] 블록 코딩 스테이지의 동작 예시

다른 슬롯에 삽입되는 inst 블록이 어떤 자료형을 가지는지를 이해하는 데에 도움을 주기 위해 해당 블록에 색을 입히기도 하였다. 여기서는, 'int1 + 5'가 정수형 값을 가지므로 해당 블록이 파란색이 되었고, 'int1 >= 30'이 bool 값을 가지므로 해당 블록이 노란색이 된 것을 볼 수 있다.

한편, [그림 4]의 코드에서는 정수형 변수인 'int1'에 문자열 형식의 값("skk but this is string")을 대입하는 오류를 범하고 있다. 유효하지 않은 형식을 사용하는 등 올바르지 않은 규칙이 확인되면 하단 콘솔에 해당 오류 내용을 출력하고, 코드 동작을 중단하게 된다. 또한, 원인이 되는 명령문 블록을 붉게 강조하여 사용자가 해당 부분을 고치도록 권장하였다.



[그림 4] 블록 코딩 스테이지의 타입 오류 예시

라. 세부 피드백 프로세스 구현

한편, [그림 5]와 같이 블록 코딩 내용을 텍스트로 변환하여 저장할 수 있도록 하였다. 텍스트 문서에는, 선언된 변수의 정보와 각 명령문의 구성, 각 명령문 오브젝트의 배치 위치, 다음으로 이동할 명령문의 줄 번호 등이 기재된다. 실제 명령문 블록이 동작하는 것과 비슷한 원리와 순서에 따라 텍스트 형식으로 변환이 되며, 이 내용은 피드백 알고리즘에 활용된다.

```

VAR int int1 0
ln 1 (START)
(-1000.0, 800.0) 2 -1
ln 2 [ v:int1 ] = [ [ v:int1 ] + [ 5 ] ]
(-868.6, 395.9) 3 -1
ln 3 (PRINT) v:int1
(-488.1, -3.8) 4 -1
ln 4 (IF) [ v:int1 ] >= [ 30 ]
(645.9, 372.8) 2 5
ln 5 (END)
(1000.0, -200.0) -1 -1

```

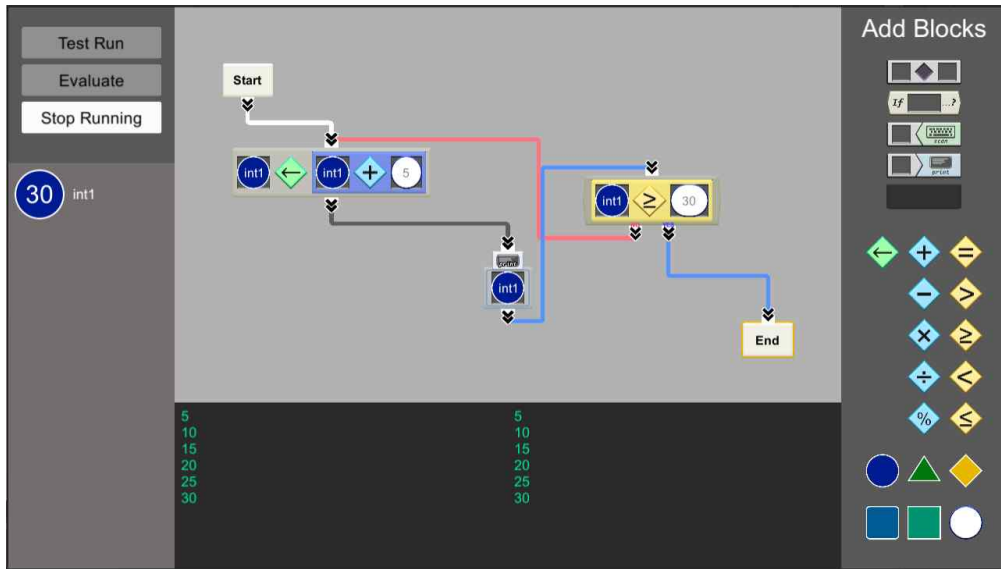
[그림 5] 블록 코딩 내용을 변환한 텍스트의 예시

[그림 5]와 같은 텍스트 변환 양식은 다음 [표 5]와 같은 규칙을 따른다.

변수 정보	VAR (변수형) (변수명) (초깃값)
명령문 정보	<p>ln (줄 번호) [명령문 내용]</p> <p>(명령문 배치 좌표) (다음 명령문 줄 번호 a) (다음 명령문 줄 번호 b)</p> <ul style="list-style-type: none"> 명령문 내용은 일반 inst 문을 제외하면 (IF), (SCAN), (PRINT), (START), (END) 등으로 구분자가 쓰인다. 명령문 블록과 변수 블록은 괄호 []로 그 내용을 표현하며, 기호 블록은 두 명령문 블록이나 변수 블록 사이에 표시된다. 다음 명령문 줄 번호는 if 명령문인 경우에만 1과 2를 모두 사용하며, if 문의 판단 기준이 False인 경우가 줄 번호 a, True인 경우가 줄 번호 b를 사용한다. 이외의 명령문에서는 줄 번호 a만 사용하며, 다음 줄 번호가 없는 경우는 그 자리에 -1을 입력한다.

[표 5] 코드 텍스트화 규칙

각 스테이지에서는 요구되는 정답이 있다. 예를 들어, [그림 6]의 상황은 “30 이하의 5의 배수를 모두 출력하는 것”을 요구하는 문제가 주어져 있던 상황이다. 작성된 코드는 ‘int1’을 5씩 늘려가며 5의 배수를 하나씩 출력하고, 이 값이 30 이상이 되면 ‘End’ 블록으로 보내는 코드이다. 왼쪽 위의 ‘Evaluate’ 버튼을 누르면 하단 콘솔 창이 두 영역으로 분리가 되면서 우측에 모범답안이 제시되는데, ‘End’ 블록에 이르기까지 출력된 내용이 이 모범답안과 일치하면 해당 문제를 올바르게 풀 것으로 인정한다.

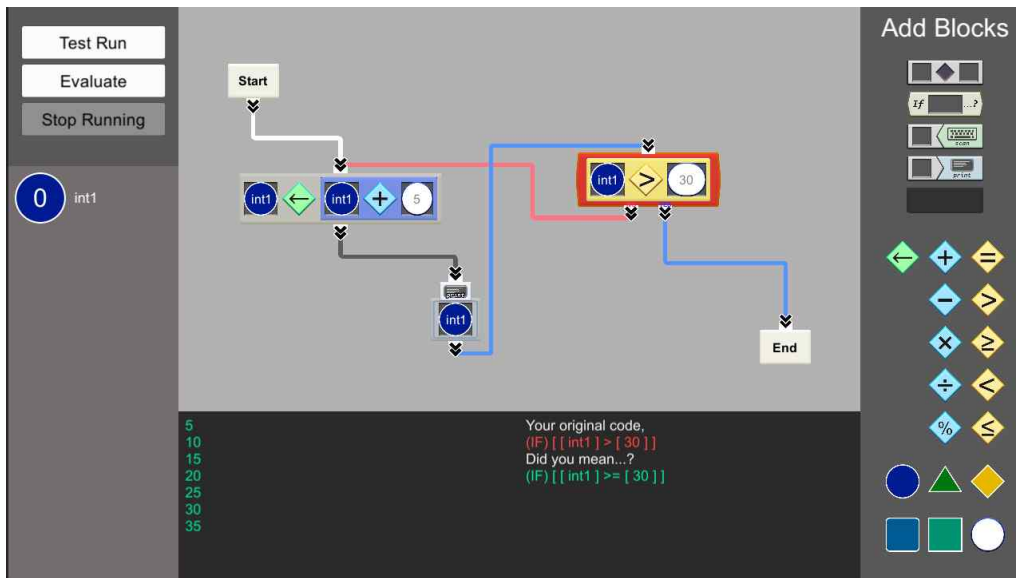


[그림 6] 작성한 코드의 출력을 정답 출력과 비교하는 과정의 예시

한편, 모범답안과 출력 내용이 일치하지 않으면 오답이 된다. 앞서 언급한 내용에서 [그림 4]와 같이 타입 오류가 감지되어 코드 실행이 중단된다면, 틀린 블록이 빨간색 표시됨으로써 사용자가 해당 블록을 고쳐야 한다는 것을 인지할 수 있다. 하지만, 별다른 오류가 뜨지 않으면서 출력 내용이 달라 오답이 발생한다면, 어느 부분이 잘못되었는지 인지하기 상대적으로 어려울 것이다. 변수의 상태 변화를 실시간으로 보면서 잘못된 점을 파악하도록 하는 것이 이 애플리케이션의 설계 방향이지만, 좀 더 구체적으로 피드백을 줄 필요도 있었다.

이때 [그림 5]의 텍스트 정보를 활용하여 피드백을 제시할 수 있다. 각 문항에 대해 모범답안이 되는 코드를 [그림 5]와 같은 형식으로 저장을 해두고, 사용자가 실제로 작성한 코드의 텍스트 변환 결과와 비교를 하는 방식이다.

‘Evaluate’ 과정에서 오답이 발생했으나 형식 오류 등 별다른 문제가 없었다면, 내부적으로 작성된 텍스트형 코드의 각 줄의 내용을 정답 코드의 내용과 비교하면서 어떤 줄이 정답 코드의 내용에 대응되는지를 텍스트 비교를 통해 정리한다. [그림 7]은 [그림 6]의 예시에서 if 명령문의 내용만 변형하여 채점을 실행한 결과이다. ‘>=’ 대신 ‘>’를 삽입하는 바람에 ‘35’가 출력이 된 상황이다. 이때, 내부적인 테스트 매핑에 따라 정답 코드가 이 부분에서 ‘>=’를 제안하고 있었다는 점을 우측 콘솔에 출력하고, 해당 블록의 위치를 찾아 붉게 표시함으로써 사용자에게 피드백을 남겨줄 수 있다.



[그림 7] 정답 코드와 비교한 내용을 바탕으로 대안을 제시하는 예

마. 게임 편의성 확보를 위한 요소

사용자의 시점에서 게임을 쾌적하게 즐길 수 있어야 하므로, 테스트 동작을 수행하며 불편했던 부분들을 개선하기도 했다. 이 부분은 구체적으로 기획하면서 구현했다기보다는, 연구 과정 도중에 가능하다고 판단되는 부분을 즉흥적으로 구현한 것이라고 볼 수 있다. 그중 몇 가지 내용은 다음과 같다.

(1) 명령문 연결선 최적화

- Unity의 Line Renderer 기능을 활용하였다.
- node를 4개 두어 일직선으로 연결했을 때보다 가시성이 좋아지도록 하였으며, 선이 자신의 블록을 직접 침범하지 않도록 간단한 거리 계산 알고리즘을 설계했다.
- 선이 뻗어나가는 방향이 위인지 아래인지에 따라 node의 수평 이동 혹은 수직 이동 여부가 결정되며, 최종적인 선의 형태는 직각 'Z'자가 되도록 하였다.

(2) 블록 크기 최적화

- 슬롯에 삽입되는 블록이나 그 내부 텍스트의 양에 따라 블록 크기를 최적화한다.
- 슬롯에 블록이 장착되거나 해제되는 동작이 발생할 때마다 관련된 블록에 관해

가장 내부에 장착된 블록부터 블록 크기를 계산하여 알맞은 블록 크기로 변형하도록 하였다.

바. 스테이지 설계를 위한 요소

스테이지 형식의 설계를 위해서는, Unity에서 지원하는 Serialize 기능을 수월하게 활용할 수 있는 json 형식의 간단한 데이터베이스 양식이 활용되었다. 각 스테이지의 순번이나 제목은 물론, 요구되는 문제나 정답에 관한 정보 또한 json으로 관리된다. 해당 스테이지를 통과했는지에 관한 정보와 같이 사용자의 플레이 데이터에 해당하는 부분도 여기서 응용할 수 있었다.

5. 결론 및 소감

가. 결론 및 개선 방향

본 연구에서는 블록 코딩 형식의 애플리케이션을 개발하였고, 이 애플리케이션을 통해 코딩을 처음 배우는 학습자들이 비교적 친숙하게 코딩을 학습할 수 있도록 도움을 주고자 하였다.

이 애플리케이션의 블록 코딩에서는 명령문에 해당하는 블록을 연결선으로 잇는 방식을 채택하여 알고리즘의 진행 순서에 관한 고민을 학습자들이 체험해볼 수 있는 환경을 제공하고자 하였다. 다소 진부하게 느껴질 수 있는 줄코딩 학습의 단점을 ‘나만의 길’을 설계하는 방식의 게임으로 보완할 수 있을 것으로 기대하였다.

아울러, int, float 등과 같은 정수와 실수의 구분, char과 string으로 표현되는 문자 혹은 문자의 집합, 진위 판별을 위한 bool 자료형 등 기초적인 영역에서의 자료 구조에 관한 학습도 도모할 수 있을 것으로 보인다. 순서도를 통한 알고리즘의 이해가 선행되면, 학습자들이 이후 학습 단계에서 심화 자료형 학습을 더욱 수월하게 공부할 수 있을 것으로 기대하였다.

또한, 피드백 시스템을 설계해보며 사용자의 관점에서 어떤 피드백이 이루어져야 효과적인 학습이 가능할지 직접 고민해볼 수 있었다. 구현한 블록 코딩 시스템에 알맞은 형태로 형식 오류나 실행 오류 등을 표시함으로써 사용자가 문법 이해를 하도록 하고, 정답을 제시하는 코드 피드백 시스템을 통해서도 사용자가 직접적인 오류와 관련 없는 오답이 어디서 발생하는지 파악하는 데에 도움을 주고자 하였다.

조금 더 정제되고 체계화된 시스템이 갖추어진다면 기획한 부분들은 예상한 기대효과를 보일 수 있을 것으로 생각한다. 그러나, 연구 수행 기간이나 규모 등을 고려했을 때 부족하게 느껴지는 부분이나 후속 연구로 개선되면 좋을 내용도 있었다.

첫째로, 게임으로서의 완성도가 다소 미흡하다는 점이다. 시스템적인 구현에 중점적으로 신경을 쓴 연구인 만큼, 그래픽의 디테일이 정교하지는 않은 편이며 오디오 효과나 애니메이션 등 게임성을 높일 요소를 활용할 여지가 매우 부족했다. 만약 비슷한 프로젝트가 실무적인 영역이나 상업적인 영역에서 진행되었다면 적극적으로 보완해보고 싶었을 것 같다.

둘째로, 실제 학습 효과에 관해서 학습자의 경험을 수집하기가 어려웠다는 점이다. 코딩 교육 프로그램으로서의 틀은 갖추었지만, 기획된 부분을 구현한 이후 사용자에게 직접 피드백을 받으며 개선점을 반영할 기회가 매우 드물었다는 점이 아쉬움에 남

는다. 코드 피드백 시스템에 관해서는 본 연구작품 수행과는 별개로 개인적으로 더 연구를 해보고 싶다는 생각이 들기도 했다.

나. 연구 진행 소감

개인 프로젝트에 참여하면서 쌓았던 Unity 게임 엔진에 관한 경험을 개인적으로 흥미를 느끼던 교육 분야에 접목하여 이 주제를 제안했다. 프로젝트 기획 단계부터 직접 구상을 해보며 실제 코딩에 이르는 절차를 이번 연구를 통해 다시금 직접 밟아보니 감회가 새롭기도 하였다. 대부분 프로젝트가 그렇겠지만, ‘개발 목표’를 세우고 그 목표를 ‘일정한 기한 내’에, ‘계획한 방식’으로 구현하는 것이 결코 단순한 과정이 아니다. 타 프로젝트에서도 어려움에 직면하면서 스트레스를 심하게 받아본 경험이 있어서 짜임새 있는 기획이 중요하다는 것을 항상 느끼고 있었다. 그래서 목표 수립부터 직접 내 손으로 다시 해보는 이 연구과제가 상당히 도전적이었다.

기획 단계에서는 필요한 것과 필요하지 않은 것을 명확히 구분하는 것이 중요했다. 향후 일정에 차질을 빚지 않기 위해 무엇을 할 것인지를 확실히 하는 것을 가장 중요하게 생각해보았고, 그 부분에 관해서 나 자신조차도 순간순간 스쳐 지나가는 생각이 많이 달라지는 변덕쟁이라는 것을 잘 알고 있어서 기획 내용을 문서로 만드는 것(형식적인 문서가 아니더라도 최소한 간단한 메모를 하는 정도)에 신경을 많이 기울였다. 무리하게 무언가를 구현하고 싶다는 욕심이 차오를 때 그걸 억제할 수 있는 확실한 수단이 문서이기도 했고, 목표가 아무리 방대하더라도 할 수 있는 일 안에서 하는 것이 가장 중요하다고 생각했기 때문이다.

실제 개발 단계는 친숙함 속에서 새로운 것을 만나는 일의 연속이었다. 타 프로젝트에서 Unity 엔진을 활용해 게임을 직접 개발해보고 대중 앞에 공개해본 경험도 있었지만, 게임의 성격상 게임 오브젝트의 접합이나 충돌과 같은 것을 직접 다룰 기회가 많지 않았기 때문이다. 그래서 변수 오브젝트나 기호 오브젝트 등을 슬롯에 직접 드래그 앤드 드롭으로 갖다 놓는 시스템을 직접 구현하는 것이 새롭게 느껴지기도 했다. 물론, 이러한 기능을 객체지향 프로그래밍에 비교적 어렵지 않게 응용할 수 있었던 것은 타 프로젝트에서 경험으로 쌓아온 것이 꽤 있었기 때문이라고 생각한다. 그래서 다양한 도전을 겪어보는 것의 중요성도 실감할 수 있었던 것 같다.

6. 참고문헌

- [1] 임수현, 정은선. "코로나19 기점으로 나타난 초등학교 4, 6학년 수학 학업성취도 변화 분석" 한국초등교육 32.3 pp.249-266 (2021) : 249.
- [2] 한성민. (2022). COVID-19가 학습 및 생활에 미친 영향: 초등학생 중심으로. 응용경제, 24(2), 55-85.
- [3] 최지선, 상경아. (2019). 초등학생 수학 성취도에 영향을 미치는 교육맥락변인에 대한 동아시아 5개국 비교. 초등수학교육, 22(3), 167-180.
- [4] 백영균, 정용석. (2004). 게임기반학습에서 학습자의 게임능력 및 학습능력이 논리적사고력에 미치는 효과. 교육정보미디어연구, 10(4), 119-140.
- [5] N. Kim, "A Study on the effect of coding education and improvement of learning achievement using educational game," Journal of Korea Game Society, vol. 17, no. 4. Korea Academic Society of Games, pp. 161-168, 31-Aug-2017.
- [6] 안성혜. "유아용 코딩아트 교육을 위한 커리큘럼 설계" 예술교육연구 17, no.2 (2019) : 43-60.
- [7] S.-J. Kim and D.-E. Cho, "A Study on Learning Model for Effective Coding Education," Journal of the Korea Convergence Society, vol. 9, no. 2, pp. 7-12, Feb. 2018.
- [8] G. Tisza and P. Markopoulos, "Understanding the role of fun in learning to code," International Journal of Child-Computer Interaction, vol. 28. Elsevier BV, p. 100270, Jun-2021.

- [9] O. Karram, "The Role of Computer Games in Teaching Object-Oriented Programming in High Schools - Code Combat as a Game Approach," WSEAS TRANSACTIONS ON ADVANCES in ENGINEERING EDUCATION, vol. 18. World Scientific and Engineering Academy and Society (WSEAS), pp. 37-46, 02-Apr-2021.
- [10] Y. Hu, U. Z. Ahmed, S. Mechtaev, B. Leong and A. Roychoudhury, "Re-Factoring Based Program Repair Applied to Programming Assignments," 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA, 2019, pp. 388-398.