JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 31 No. 2, pp. 249-257, February 2026
https://doi.org/10.9708/jksci.2026.31.02.249

Analysis of Errors and Misconceptions in Scratch Programming among Pre-Service Elementary Teachers

Junghee Jo*

*Associate Professor, Dept. of Computer Education, Busan National University of Education, Busan, Korea

[Abstract]

This study aims to analyze errors observed in Scratch programming learning among pre-service elementary teachers and to identify the characteristics of misconceptions underlying those errors. A 15-week Scratch course was conducted with 44 participants, and error data were collected using 22 assessment items covering 11 core concepts. The results show low error rates in intuitive functions such as sprite movement and coordinate setting, but higher error rates in concepts requiring an understanding of logical execution flow and state changes, including variables, conditionals, loops, and events. Recurrent misconceptions were observed regarding the timing of variable assignment, independent execution of conditionals, and evaluation of loop termination conditions. These findings are expected to be meaningful in that they can be utilized as foundational evidence for the future design of error- and misconception-based curricula or the development of instructional and learning models.

▸ **Key words**: Scratch, Pre-service elementary teacher, Error, Education, Programming

[요 약]

본 연구는 예비초등교사를 대상으로 한 스크래치 프로그래밍 학습에서 나타나는 오류를 분석하고, 오류에 내재된 오개념의 특성을 규명하는 데 목적이 있다. 이를 위해 교원양성대학에 재학 중인 예비초등교사 44명을 대상으로 15주간 스크래치 교육을 실시한 후, 핵심 개념 11개 영역에서 총 22개의 평가 문항을 개발하여 오류 데이터를 수집·분석하였다. 연구 결과, 스프라이트 이동이나 좌표 설정과 같은 직관적인 기능에서는 낮은 오류율을 보였으나, 변수, 조건문, 반복문, 이벤트 등 프로그램의 논리적 실행 흐름과 상태 변화를 요구하는 개념에서는 높은 오류율이 나타났다. 또한 변수 대입의 실행 시점, 조건문의 독립적 실행, 반복문의 종료 조건 평가 시점과 관련된 공통된 오개념이 반복적으로 관찰되었다. 이러한 결과는 향후 오류·오개념 기반 교육과정 설계나 교수·학습 모형 개발을 위한 근거 자료로 활용될 수 있다는 점에서 의의를 지닐것으로 기대된다.

▸ **주제어**: 스크래치, 예비초등교사, 오류, 교육, 프로그래밍

• First Author: Junghee Jo, Corresponding Author: Junghee Jo
*Junghee Jo (dreamer@bnue.ac.kr), Dept. of Computer Education, Busan National University of Education
• Received: 2025. 12. 16, Revised: 2025. 12. 27, Accepted: 2026. 01. 20.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

2022 개정 교육과정 총론에서는 인공지능 기술 혁신에 대응할 핵심 역량을 기르기 위해 디지털 소양 강화와 정보 교육 확대를 주요 개정 방향으로 제시하고 있다. 초등학교의 경우, 실과 교과 내 정보 영역의 시수와 학교 자율시간을 활용하여 정보교육을 34시간 이상 운영하도록 명시함으로써 프로그래밍 교육이 중·고등학교뿐만 아니라 초등 단계에서도 체계적인 교육과정의 핵심 요소로 강조되고 있다[1].

프로그래밍 교육의 비중이 확대되면서 초등학습자들은 보다 이른 시기에 프로그래밍을 경험하게 되었다. 초등 교육 현장에서는 스크래치(Scratch) 또는 엔트리(Entry)와 같은 블록형 언어를 활용한 프로그래밍 교육이 주로 이루어지고 있다. 블록 기반의 프로그래밍 환경은 문법적 부담을 최소화하여 초기 학습자의 진입 장벽을 낮추고, 시각적 피드백을 통해 학습 동기를 향상시키는 장점을 가진다[2]. 그러나 이러한 블록형 프로그래밍 언어를 활용한 소프트웨어 개발 과정에서도 오류는 필연적으로 발생하며, 경험이 부족한 초보 학습자는 이러한 오류를 전문가의 도움 없이 스스로 해결하기 어려운 부분이 있어서 이는 프로그래밍 교육에서 핵심적으로 다루어야 할 문제로 인식되고 있다[3-6].

프로그래밍 오류에 관한 연구는 오랜 역사를 지니며, 주로 C나 Python과 같은 텍스트 기반 프로그래밍 언어를 중심으로 축적되어 왔으나 블록형 프로그래밍 언어를 사용하는 학습 환경에서 발생하는 오류를 다룬 연구는 아직 초기 단계에 머물러 있다. 최근 프로그래밍 교육이 초등교육과정의 일부로 포함되면서, 블록형 프로그래밍 환경에서 나타나는 오류뿐만 아니라 학습자의 오개념을 탐색하려는 연구들이 점차 이루어지고 있으나, 그 수와 범위는 여전히 제한적인 실정이다. 프로그래밍 오개념(Misconception)이란 프로그래밍 개념 또는 그와 관련된 개념을 잘못 이해하는 것을 의미하며, 이러한 오개념은 특히 프로그래밍 입문 단계에서의 학습 실패를 초래하고 장기적으로는 학습자가 프로그래밍 학습을 중도에 포기하게 만드는 주요 원인 중 하나로 보고되고 있다[8].

본 연구에서는 교원양성대학에 재학 중인 예비 초등교사를 대상으로 스크래치를 학습시킨 후, 선행연구를 기반으로 스크래치에서 자주 발생하는 오류를 중심으로 평가 문항을 구성하였다. 또한 특정 문항을 해결하는 과정에서 반복적으로 관찰되는 오답 패턴을 분석하여 예비교사들의 학습 실패를 초래하는 오개념을 도출하였다. 그리고 이를 통해 향후 스크래치 교육에서 이러한 오류를 사전에 예방하고 지도 방안을 개선하는 데 필요한 시사점을 제시하였다.

II. Related Works

블록 기반 프로그래밍 환경에서 나타나는 학습자의 오류는 단발적인 실수에 그치기보다, 개념적 오해와 밀접하게 연결될 수 있다. 특히 스크래치와 같은 시각적 프로그래밍 언어는 문법 오류를 크게 감소시키는 장점이 있는 반면, 학습자가 프로그램의 실행 흐름, 상태 변화, 조건 평가, 이벤트 동기화와 같은 비가시적인 계산 과정을 충분히 이해하지 못할 경우, 이러한 이해 부족이 반복적인 오답 패턴으로 드러나는 경향이 있다. Grover와 Basu는 초급 블록 프로그래밍 학습을 위해 반복문, 변수, 논리식과 관련된 문항을 설계하고 학습자의 오답을 분석한 결과, 동일한 유형의 오답이 지속적으로 발생하는 경우 이를 개념적 오해, 즉 오개념으로 해석할 수 있음을 제시하였다[9].

오개념 연구는 전통적으로 과학 및 수학 교육 분야에서 학습자의 대안적 개념을 진단하고 효과적인 교수·학습 전략을 설계하기 위한 이론적인 틀로 활용되어 왔다[10]. 프로그래밍 맥락에서의 오개념은 반복, 조건, 변수, 동기화와 같은 핵심 개념을 부분적으로만 이해하거나 일상적 직관에 기반하여 잘못 일반화함으로써, 표면적으로는 정답과 유사한 사고 과정을 보이지만 결과적으로는 오류를 유발하는 방식으로 나타난다. 이러한 오개념은 단발성 실수와는 달리, 문항이나 과제의 맥락이 달라지더라도 일정한 형태의 오답 패턴으로 반복 재현된다는 점에서 구별된다[11-13]. Swidan은 스크래치 환경에서 나타나는 대표적인 오개념으로 명령문의 순차성에 대한 이해 부족, 변수가 동시에 하나의 값만 가질 수 있다는 개념에 대한 오해, 그리고 사용자 입력이 요구되는 상황에서 프로그램의 상호작용성에 대한 이해 부족을 제시하였다[7].

국내에서도 블록 기반의 교육용 프로그래밍 언어 학습자의 어려움과 개념적 오류를 탐색한 연구들이 점차 보고되고 있다. 선행연구에 의하면, 초보 학습자들은 리스트 제어의 의미를 충분히 이해하지 못하거나 변수와 함수의 필요성을 과소평가하는 경향을 보이며, 프로그램에서 발생한 논리적 오류의 원인을 스스로 파악하는 데에도 어려움을 겪는 것으로 나타났다[14]. 특히 예비교사 집단은 현재로서는 프로그래밍 초보자이지만, 향후 초등교육 현장에서 지도자의 역할을 수행해야 한다는 점에서 오개념 연구의 의미가 크다. 이에 따라 예비교사를 대상으로 한 블록 프

로그래밍 오류 분석 연구도 점차 이루어지고 있다.

예비교사의 경우, 단기간에 블록을 조합하여 산출물을 생성하는 수행은 가능하더라도, 실제 수업 현장에서 설명과 피드백을 제공할 수준의 개념적 정교화가 충분히 이루어지지 않을 가능성이 있다. 이러한 문제의식을 바탕으로 최근 연구에서는 예비교사의 스크래치 명령어 이해 과정에서 나타나는 오류를 단서로 오개념을 탐색하고, 테스트 기반 활동을 통해 개념을 재구성하도록 하는 교수·학습 방법들이 제안되고 있다[3,4]. 이는 예비교사 교육에서 오개념을 조기에 진단하고 교수 전략을 통해 수정하는 과정이 중요함을 시사한다. 그러나 이러한 연구는 국외에 비해 국내에서는 여전히 제한적으로 이루어지고 있는 실정이다.

이에 본 연구에서는 국내 예비 초등교사를 대상으로 스크래치 평가 문항에서 반복적으로 관찰되는 오답 패턴을 수집·분류하고, 이를 토대로 개념별 오개념을 도출함으로써 예비교사 대상 블록 프로그래밍 교육에서 오류 예방 및 지도 방안 개선을 위한 시사점을 제시하고자 한다.

III. Research Design

1. Subject

본 연구는 교원양성대학 1학년에 재학 중인 예비 초등교사 44명을 대상으로 수행되었다. 이 중 남학생은 16명(36.4%), 여학생은 28명(63.6%)이었다.

Table 1. Participants' Prior Experience with Programming Languages

Programming Experience	Number(%)
Block-based	30 (68.2%)
Text-based	16 (36.4%)
Either	31 (70.5%)
Both	15 (34.1%)
None	13 (29.5%)

Table 1에 제시된 바와 같이, 전체 예비교사 중 블록 프로그래밍 또는 텍스트 프로그래밍 중 하나 이상을 경험한 예비교사는 70.5%였으며, 두 유형 모두를 경험하지 못한 예비교사는 29.5%였다. 또한 블록 프로그래밍 경험자 비율(68.2%)은 텍스트 프로그래밍 경험자 비율(36.4%)의 거의 두 배에 달하였고, 두 유형을 모두 경험한 예비교사는 전체의 34.1%로 나타났다.

블록 프로그래밍을 경험한 예비교사는 모두 스크래치 또는 엔트리를 학습한 것으로 나타났으며, Table 2에 제시된 바와 같이 스크래치 경험자(50%)보다 엔트리 경험자(73.3%)의 비율이 더 높았다. 또한 두 유형을 모두 경험한 예비교사는 전체 블록 프로그래밍 경험자의 20%를 차지하였다.

Table 2. Types of Programming Languages

Profile	Category	Number(%)
Block-based	Scratch	15 (50.0%)
	Entry	22 (73.3%)
	Both	6 (20.0%)
Text-based	Python	13 (81.3%)
	C	3 (18.8%)
	Java	1 (6.3%)

텍스트 프로그래밍을 경험한 예비교사는 Python, C, 또는 Java를 학습하였으며, 이 중 Python 경험자가 81.3%로 가장 높은 비중을 보였다. 반면, C와 Java 경험자는 각각 18.8%와 6.3%에 불과하였다.

2. Curriculum

예비 초등교사는 프로그래밍 영역에서 비전공자로서 대부분 초보자 수준에 머무르며, 동시에 향후 초등교육 현장에서 교수자의 역할을 수행해야 한다는 이중적 특성을 지닌다. 2022 개정 교육과정의 소프트웨어 교육 영역은 문제 해결을 위한 알고리즘 설계, 프로그램 실행 과정의 이해, 오류 수정과 개선을 핵심 성취기준으로 제시하고 있다. 이러한 교육과정의 요구를 반영하여 본 강의는 스크래치의 핵심 개념 이해를 토대로, 초등 학습자가 프로그램의 실행 흐름과 논리를 이해하지 못해 발생시키는 오류를 예측·분석하고 이를 지도할 수 있는 역량을 기르는 데 중점을 두어 구성되었다.

강의는 15주에 걸쳐서 진행되었으며 세부적인 강의 내용은 Table 3과 같이 구성하였다. 우선, 스프라이트의 개념과 사용법을 소개하고, 스프라이트 간 상호작용을 제어하는 방법을 실습하였다(Use of sprites). 그리고 스프라이트의 움직임을 이해하기 위해 좌표, 방향, 회전 개념을 학습하였다(Orientation). 다음으로 사칙연산, 관계연산, 논리연산 등 다양한 연산 블록의 활용을 익혔으며(Operations), 데이터 저장과 변화 추적을 위한 변수 개념을 학습하였다(Variables).

Table 3. The Curriculum Used for the Lecture

Category	Contents
Survey	Programming background
Curriculum	Use of sprites
	Orientation
	Operations
	Variables
	Conditional statements
	Loops
	Events
	Sensing
	Lists
	Functions (My Blocks)
	Machine Learning for Kids (AI)
Evaluation	Final examination

다음으로, 조건문(Conditional statements)과 반복문(Loops)을 통해 프로그램의 흐름 제어 방식을 학습하였고, 이벤트 블록을 활용하여 스프라이트 간 동기화와 상호작용을 구현하였다(Events). 그리고 감지 블록을 활용하여 사용자의 입력에 반응하는 프로그램을 제작하였다(Sensing).

강의 후반에는 리스트 블록을 통해 다수의 데이터를 처리하는 방법을 학습하였고(Lists), '내 블록'을 이용한 함수 개념을 적용하여 코드의 재사용성과 구조화를 강조하였다(Functions). 마지막으로 IBM에서 개발한 머신러닝포키즈(Machine Learning for Kids)를 활용하여 스크래치와 인공지능을 결합한 프로젝트를 학습하였고 기말 평가를 수행하였다.

3. Evaluation and Analysis

본 연구에서 사용한 평가 문항은 스크래치 학습의 핵심 개념을 측정하기 위해 개발되었다. 각 문항은 특정 개념 요소를 중심으로 설계되었으며, 문항별로 측정하고자 하는 개념을 사전에 정의하였다. 총 문항 수는 22문항으로, 스크래치의 핵심 개념을 Table 3과 같이 총 11개 영역으로 구분하고, 각 개념 영역마다 2문항씩을 출제하여 평가를 구성하였다.

이는 단일 문항만으로는 특정 개념에 대한 학습자의 이해 수준을 안정적으로 판단하기 어렵다는 점을 고려하여, 동일 개념을 서로 다른 맥락에서 반복 측정하기 위해서였다. 두 문항은 동일 개념을 측정하되 문항 맥락과 제시 방식에 차이를 두어, 학습자가 특정 문항 형식에 익숙해져 정답을 선택하는 효과를 최소화하고 개념 이해 여부를 보다 타당하게 평가하고자 하였다.

문항은 프로그램 실행 과정에 대한 개념적 이해를 평가하기 위해 실행 결과 예측형과 오류 원인 진단형으로 구성하였으며, 초보자 대상의 블록 프로그래밍 학습 과정에서 빈번하게 나타나는 오류 유형을 문항에 반영하고자 선행 연구를 토대로 문항을 설계하였다. 이를 위해, 162가지의 대표적인 스크래치 오류를 분석하여 11개의 주요 오개념을 제시한 Swidan[7]의 연구와, 74,830개의 스크래치 프로젝트를 분석하여 오류 패턴을 도출한 Greifenstein[5]의 연구를 참고하였다.

채점은 사전에 정의된 채점 기준에 따라 이루어졌으며, 채점 결과를 기반으로 분석을 위해 핵심 개념 11개 항목 각각에 대한 오류율을 산출하였다. 각 '핵심 개념별 오류율'은 해당 영역에 속한 2문항 중 1문항 이상에서 오답을 보인 경우를 오류 발생으로 처리하여 산출하였다. 이는 특정 개념에 대해 일관된 이해가 형성되지 않은 경우, 문제 상황이나 표현 방식이 달라질 때 오류가 재현될 가능성이 높다는 오개념 연구의 관점을 반영한 것이다. 즉, 동일 개념을 측정하는 두 문항 중 하나라도 오답을 보인 경우, 해당 개념에 대한 이해가 불완전하거나 개념적 오해가 존재할 가능성이 있다고 판단하였다.

아울러 본 연구에서는 예비교사들이 특정 핵심 개념을 전반적으로 이해하고 있는지 여부뿐만 아니라, 해당 개념을 구성하는 세부 기능 요소 중 어느 부분에서 어려움을 겪고 있는지를 보다 정밀하게 파악하고자 각 '문항별 오류율'을 함께 산출하였다. 동일한 개념을 측정하는 문항이라 하더라도, 문항에 포함된 실행 맥락이나 요구되는 인지 과정에 따라 학습자의 반응 양상이 달라질 수 있기 때문이다. 문항별 오류율 분석을 통해 특정 개념 영역 내에서도 학습자의 오류가 집중되는 세부 기능이나 사고 과정이 무엇인지 식별할 수 있으며, 이는 단순한 개념 이해 여부를 넘어 예비교사의 개념 이해 구조를 보다 세밀하게 진단할 수 있다고 판단하였다.

IV. Results

1. Error Rates

본 연구에서 수집된 오류를 분석한 결과는 Table 4와 같다. Concepts열은 스크래치의 핵심 개념 11개 항목을 의미하며 Error Rate(Concepts)열은 각 핵심 개념별 오류율을 의미한다. 핵심 개념별 오류의 분석 결과, Use of sprites(4.5%), Orientation(22.7%), Operations(20.5%)과 같이 스프라이트 조작 중심의 기초 개념에서는 비교적

Table 4. Summary of Error Rates by Scratch Concepts and Question-Specific Features(n=44)

Concepts	Error Rate (Concepts)	Question-specific Features	Error Rate (Question-specific Features)
Use of sprites	2(4.5%)	Controlling interactions between sprites	1(2.3%)
		Next costume	1(2.3%)
Orientation	10(22.7%)	Coordinates	1(2.3%)
		Rotation	9(20.5%)
Operations	9(20.5%)	Simple logical expression	0(0.0%)
		Complex logical expression	9(20.5%)
Variables	38(86.4%)	Set my variables	38(86.4%)
		Change my variables	2(4.5%)
Conditional statements	36(81.8%)	Block execution order in if-then statements	36(81.8%)
		Block execution order in if-then-else statements	3(6.8%)
Loops	37(84.1%)	Repeat until in simple form	35(79.5%)
		Repeat until in complex form	18(40.9%)
Events	36(81.8%)	Understanding the basic concepts of broadcast and receive messages	36(81.8%)
		Understanding scripts that use broadcast and receive messages	3(6.8%)
Sensing	33(75.0%)	Ask and wait	13(29.5%)
		Key pressed	30(68.2%)
Lists	4(9.1%)	Add, delete	2(4.5%)
		Insert, replace	2(4.5%)
Functions (My Blocks)	22(50.0%)	The simple invocation order	6(13.6%)
		The complex invocation order	22(50.0%)
Machine Learning for Kids(AI)	31(70.5%)	Project workflow	29(65.9%)
		Improving model accuracy	11(25.0%)

낮은 오류율을 보였다. 이는 예비교사들이 스프라이트 이동, 좌표 설정, 간단한 연산과 같은 표면적, 직관적인 기능의 수행에는 큰 어려움이 없음으로 해석될 수 있다.

반면, 프로그램의 논리적 실행 흐름과 상태 변화를 이해해야 하는 고차원적 스크래치 개념인 Variables(86.4%), Conditional statements(81.8%), Loops(84.1%), 그리고 Events(81.8%)의 오류율은 매우 높게 나타났다. 이것은 단순 블록 조작 이상의 인지적 노력이 요구되는 개념으로써 예비교사들이 프로그램 구조를 구성하는 핵심 논리를 충분히 이해하지 못하고 있음으로 해석될 수 있다.

Question-specific Features 열은 11개의 핵심 개념 각각을 구성하는 두 가지 세부 기능 요소를 반영한 총 22개 문항을 의미하며, Error Rate(Question-specific Features) 열은 각 문항별 오류율을 나타낸다. 문항별 오류를 분석한 결과, 동일한 개념 내에서도 문항 간 오류율 편차가 뚜렷하게 나타났다. 예를 들어, Orientation 개념에 속한 문항 중 Coordinates 관련 문항은 2.3%의 낮은 오류율을 보였으나, Rotation 관련 문항은 20.5%로 다소 높은 오류율을 기록하였다. 이는 예비교사들이 좌표 이동과 같은 정적 개념은 비교적 잘 수행하나, 회전에 포함된 상대적 방향성이나 방향 전환과 같은 개념적 요소에는 어려움을 겪고 있음을 알 수 있다. 또한 Variables 영역에서는 변수의 기본 저장 기능에 대한 이해는 비교적 높았지만, 변수 값을 조건식과 결합하여 판단하는 문항에서는 79.5% 이상의 높은 오류율이 나타났다. 이는 변수에 대한 조작 능력은 있으나, 해당 값이 프로그램의 논리 구조에 어떤 영향을 미치는지 이해하는 과정에서 오개념이 발생함을 보여준다.

2. Misconceptions from Students' Answers

Question-specific Features별 오류율을 도식화한 결과는 Fig. 1과 같다. 총 22개의 스크래치 세부 기능 중 오류율이 가장 높은 상위 5개 항목을 대상으로, 해당 문항을 오답으로 처리한 예비교사의 사고 과정을 분석하였다. 이를 위해 평가지에 기록된 풀이 과정, 선택한 응답, 실행 결과에 대한 서술을 수집하고, 학습자가 예측한 실행 결과와

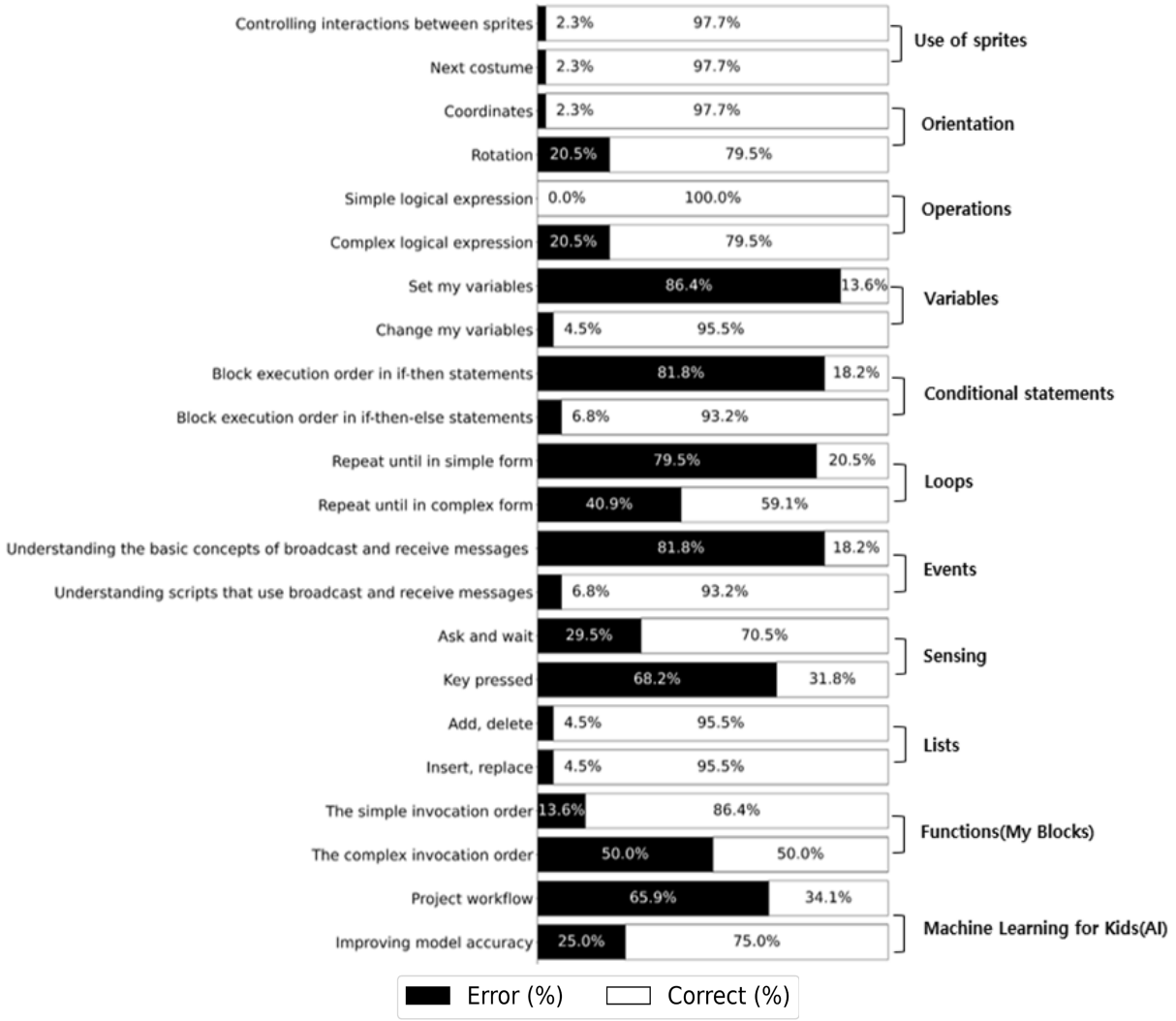

Fig. 1. Error Rates by Question-specific Features (n=44)

실제 프로그램의 실행 결과를 비교하였다. 이러한 비교를 통해 실행 흐름, 조건 판단, 변수 값 변화 등에 대한 오해가 드러나는 지점을 확인하였으며, 동일하거나 유사한 사고 과정이 반복적으로 관찰되는 경우 이를 해당 개념과 관련된 오개념으로 해석하였다.

• Set my variables

본 문항은 스크래치의 '변수' 카테고리의 '정하기' 블록의 이해도를 묻는 문항으로 44명 중 38명(86.4%)이 오답을 제출하였다. 이 문항은 Fig. 2와 같으며 Swidan[7]의 연구에서 도출된 11개의 대표적인 스크래치 오개념 중에서도 가장 오류가 많다고 보고되었다.

이 문항을 틀린 38명의 예비교사 중 19명(50%)은 2를, 19명(50%)은 6을 오답으로 제출하였다. 다수의 연구대상자가 정답인 0이 아닌 6 또는 2로 응답한 것은, 스크래치에서 변수 대입이 순차적으로 즉시 실행된다는 점을 충분히 이해하지 못했기 때문으로 해석된다.

Fig. 2. Question-specific Features:Set my variables

즉, 6으로 응답한 경우에는 합계에 설정된 계산식이 이후 변수 값의 변화에 따라 자동으로 갱신된다고 오해하여

숫자1에 대입된 값 4와 2를 모두 반영한 결과를 도출한 것으로 보인다. 반면, 2로 응답한 경우에는 스크립트의 마지막에 위치한 말하기 블록을 기준으로 마지막에 대입된 값이 출력 결과에 반영된다고 판단한 것으로 해석된다. 이러한 결과는 예비교사들이 변수의 대입 시점을 비롯하여 실행 순서를 정확히 이해하지 못하고 있음을 보여준다.

• Block execution order in if-then statements

본 문항은 스크래치의 '제어' 카테고리의 '만약~라면' 블록의 실행순서를 묻는 문항으로 '만약~라면 아니면' 블록과의 차이를 예비교사들이 올바르게 이해하고 있는지를 검증하는 것으로 총 44명 중 36명(81.8%)이 오답을 제출하였다.

Fig. 3의 실행 순서는 1-2-3-4-6임에도 불구하고 이 문항을 틀린 36명의 예비교사들 중에서 무려 30명(83%)이 1-2-3으로 응답하였다. 이것은 스크래치에서 여러 개의 만약(if) 블록이 순차적으로 독립 실행된다는 점을 정확히 이해하지 못한 데에서 비롯된 것으로 해석된다. 즉, 예비교사들은 첫 번째 조건문이 참으로 평가되어 해당 블록이 실행되면 이후 조건문은 더 이상 평가되지 않거나 프로그램 실행이 종료된다고 오해한 것으로 보인다.

Fig. 3. Question-specific Features: Block execution order in if-then statements

• Understanding the basic concepts of broadcast and receive messages

본 문항은 스크래치의 '감지' 카테고리의 '신호보내기' 블록에 대한 기본적인 이해를 평가하는 사지선다형의 비교적 낮은 난이도의 문항임에도 불구하고, 총 44명 중 36명(81.8%)이 오답을 선택하였다. 이러한 결과는 다수의 예비교사들이 신호 보내기를 특정 스프라이트를 대상으로 전달되는 1:1 통신 방식으로 오해하여, 신호를 보내는 스프라이트나 신호를 받는 스프라이트의 모양 개수와 신호 전달이 논리적으로 관련된다고 판단한 것으로 보인다.

• Repeat until in simple form

본 문항은 스크래치의 '제어' 카테고리의 '~까지 반복하기' 블록의 이해도를 묻는 문항으로 44명 중 35명(79.5%)이 오답을 제출하였다.

Fig. 4. Question-specific Features: Repeat until in simple form

Fig. 4에서 '나의변수'에 최종적으로 저장된 값의 정답은 7임에도 불구하고, 해당 문항을 오답으로 처리한 35명의 예비교사 중 34명(97%)이 모두 동일하게 6으로 응답하였다. 이는 '~까지 반복하기' 블록에서 반복 종료 조건이 언제 검사되는지에 대한 강한 오개념이 공유되고 있음을 나타낸다.

예비교사들은 반복문 내부에서 변수 값이 증가한 이후에 조건이 평가된다는 실행 순서를 인식하지 못하고, 반복 종료 조건이 증가 연산 이전에 적용되는 사전 검사 방식으로 반복문을 해석한 것으로 보인다. 그 결과 조건이 참이 되는 시점의 값이 아닌, 그 직전의 값을 최종 결과로 판단하여 실제 값보다 1 작은 값을 응답하게 된 것으로 해석된다. 이는 반복문의 실행 원리에 대한 형식적 이해가 아닌, 직관적 규칙에 의존한 사고가 예비교사들의 문제 해결에 영향을 미쳤을 수 있음을 보인다.

• Key pressed

본 문항은 스크래치의 '감지' 카테고리의 '~키의 눌렀는가' 블록의 이해도를 묻는 문항으로 44명 중 30명(68.2%)이 오답을 제출하였다. Fig. 5 문항은 Greifenstein[5] 연구에서 도출한 대표적인 스크래치 오류중의 하나로써 이

러한 오류 패턴은 74,830개의 스크래치 프로젝트중에서 3,282개가 발견되었다고 보고된다.

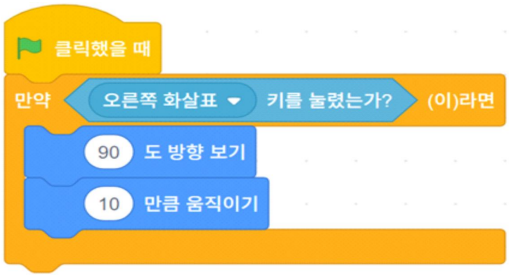

Fig. 5. Question-specific Features: Key pressed

해당 프로그램의 실행 결과는 '스프라이트는 움직이지 않는다' 임에도 불구하고, 해당 문항을 오답으로 처리한 30명의 예비교사 중 17명(57%)이 동일하게 '스프라이트는 한 번만 움직인다'로 응답하였다. 이는, 키 입력 조건이 실시간으로 반복 평가되는 이벤트 기반 구조임을 인식하지 못하고, 조건문을 일회성 분기 구조로 해석한 데에서 비롯된 오개념으로 해석될 수 있다.

V. Conclusions

본 연구는 스크래치 기반 프로그래밍 평가 문항을 대상으로 주요 개념 수준과 세부 기능 수준에서의 오류율을 분석하고, 오답 응답에 나타난 예비초등교사의 사고 과정을 검토하여 오개념의 양상을 탐색하였다. 이를 통해 다음과 같은 결론을 도출하였다.

첫째, 스크래치의 핵심 개념 11개에 대한 오류율을 분석한 결과, 스프라이트 이동, 좌표 설정, 기본 연산과 같이 비교적 직관적인 기능에서는 낮은 오류율이 나타났다. 반면에 변수, 조건문, 반복문, 이벤트 영역에서는 상대적으로 높은 오류율이 관찰되었다. 이는 프로그램의 논리적 실행 흐름과 상태 변화를 종합적으로 이해해야 하는 개념에서 예비교사들이 어려움을 겪고 있을 가능성을 나타낸다. 이러한 결과는 스크래치 교육에서 단순한 블록 사용 능력의 숙달을 넘어, 프로그램 실행 구조와 상태 변화에 대한 이해를 중심으로 한 교수·학습 전략이 더 강조될 필요가 있음을 뒷받침하는 자료로 해석될 수 있다. 예를 들어 변수 값의 변화 과정을 시각적으로 추적하는 활동, 의사코드(pseudocode)를 활용하여 조건문과 반복문의 실행 흐름을 예측하는 연습, 이벤트 메시지를 도식화하여 실행 구조를 시각적으로 표현하는 활동 등은 해당 오개념을 완화하는 교수 전략이 될 수 있다.

둘째, 총 22개 평가 문항을 대상으로 개별 기능 수준의 오류율을 분석한 결과, 동일한 개념 영역 내에서도 문항 간 오류율의 편차가 뚜렷하게 나타났다. 특히 변수 대입의 실행 시점, 조건문의 실행 순서, 반복문의 종료 조건 평가 시점, 이벤트 기반 실행 구조와 관련된 문항에서 공통적인 오류 양상이 반복적으로 관찰되었다. 또한 프로그램의 실행을 단계적으로 추적하기보다는 직관적인 규칙이나 자연어적 해석에 의존하는 경향이 일부 학습자에게서 두드러졌으며, 이로 인해 유사한 오답이 다수의 예비교사에게서 반복적으로 나타나는 양상을 보였다. 이러한 결과는 스크래치 기반 프로그래밍 교육에서 블록을 연결한 결과를 확인하는 활동도 필요하지만, 논리적 추론, 실행 흐름의 이해, 상태 변화의 추적과 같은 핵심 요소를 보다 체계적으로 다루는 활동도 중요할 수 있음을 나타낸다.

REFERENCES

[1] H.J. Choi and S.K. Shin, "A Study of Programming Education Contents in Elementary School under the National Curriculum", Journal of The Korean Association of Information Education, Vol. 29, No. 5, pp. 573-587, 2025. DOI: 10.14352/jkaie.2025.29.5.573

[2] K.M. Ahn, W.S. Sohn, and Y.C. Choy, "The Effect of Scratch Programming Education on Learning-Flow and Programming Ability for Elementary Students", Journal of The Korean Association of Information Education, Vol. 15, No. 1, pp. 1-10, 2011. UCI: G704-000854.2011.15.1.018

[3] W.S. Moon, "Analysis of error data generated by prospective teachers in programming learning", Journal of the Korean Association of information Education, Vol. 22, No. 2, pp. 205-212, 2018. DOI: 10.14352/jkaie.2018.22.2.205

[4] C.H. Lee, "Analysis of EPL Programming Errors committed by Elementary Pre-service teachers", Korean Practical Arts Education, Vol. 26, No. 3, pp. 133-150, 2020. DOI: 10.29113/skpaer.2020.26.3.007

[5] L. Greifenstein, F. Obermüller, E. Wasmeier, U. Heuer, and G. Fraser, "Effects of hints on debugging Scratch programs: An empirical study with primary school teachers in training", In Proceedings of the 16th Workshop in Primary and Secondary Computing Education, pp. 1-10, Oct. 2021. DOI: 10.1145/3481312.3481344

[6] C. Frädrich, F. Obermüller, N. Körber, U. Heuer, and G. Fraser, "Common bugs in scratch programs", In Proceedings of the 2020 ACM conference on innovation and technology in computer science education, pp. 89-95, June 2020. DOI: 10.1145/3341525.3387389

[7] A. Swidan, F. Hermans, and M. Smit, "Programming misconceptions for school students", In Proceedings of the 2018 ACM Conference on International Computing Education Research, pp. 151-159, August, 2018. DOI: 10.1145/3230977.3230995

[8] L. Ma, J. Ferguson, M. Roper, and M. Wood, “Investigating and improving the models of programming concepts held by novice programmers”, Computer Science Education, Vol. 21, No. 1, pp. 57–80, 2011, DOI: 10.1080/08993408.2011.554722

[9] S. Grover and S. Basu, “Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic”, In Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education, pp. 267-272, March, 2017. DOI: 10.1145/3017680.3017723

[10] R. Duit and D.F. Treagust, “Conceptual change: A powerful framework for improving science teaching and learning”, International Journal of Science Education, Vol. 25, No. 6, pp. 671-688, 2003. DOI: 10.1080/09500690305016

[11] Ž.Žanko, M. Mladenović, and D. Krpan, “Analysis of school students' misconceptions about basic programming concepts”, Journal of Computer Assisted Learning, Vol. 38, No. 3, pp. 719-730, 2022. DOI: 10.1111/jcal.12643

[12] M. Mladenović, Ž. Žanko, and G. Zaharija, “From blocks to text: Bridging programming misconceptions”, Journal of Educational Computing Research, Vol. 62, No. 5, pp. 1082-1106, 2024. DOI: 10.1177/07356331241240047

[13] M. Mladenović, I. Boljat, and Ž. Žanko, “Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level”, Education and Information Technologies, Vol. 23, No. 4, pp. 1483–1500, 2017. DOI: 10.1007/s10639-017-9673-3

[14] S.Y. Yoon, H.J. Choi, and S.H. Kim, "Educational Materials for Block Programming-Based Data Structures to Enhance Teacher Competence in Software Classes". Journal of Digital Contents Society, Vol. 24, No. 4, pp. 743-753, 2023. DOI: 10.9728/dcs.2023.24.4.743

Authors

Junghee Jo received the Ph.D. degree in Computer Science from the University of Massachusetts Amherst, USA, in 2014. She previously worked at LG Electronics and the Electronics and Telecommunications Research Institute (ETRI). She is currently an Associate Professor in the Department of Computer Education at Busan National University of Education, Korea. Her research interests include computer education, human factors, and health informatics.