컴퓨터교육학회 논문지 2026년 제29권 제1호
https://doi.org/10.32431/kace.2026.29.1.003

파이썬 입문자를 위한 Parsons Problems 기반의 프로그래밍 환경 개발*

Developing a programming environment based on Parsons Problems for beginners in Python

김용천† · 김자미††
Yongcheon Kim† · Jamee Kim††

요약

SW·AI 등 변화하는 세상을 이해하는 컴퓨팅사고력의 핵심인 프로그래밍은 여전히 입문자에게 높은 진입장벽으로 작용한다. 이에 본 연구는 프로그래밍에 쉬운 접근성을 제공하는 Parsons Problems의 단점을 보완하고, 유연성을 갖춘 프로그래밍 환경을 개발하기 위한 목적으로 진행되었다. 텍스트 기반 언어 학습에 블록의 활용을 통해 학습자의 수준을 고려한 맞춤형 교육을 제공하기 위한 환경을 설계하였다. 이후, 85명의 대학생을 대상으로 15주간 수업을 진행하여 설계된 환경에 대한 사용성을 평가하였다. 평가 결과, 사용 자신감, 프로그래밍 기능 이해, 유용성에서 파이썬 무경험자의 만족감이 경험자에 비해 통계적으로 유의하게 높았다. 본 연구는 텍스트 프로그래밍 입문자의 진입장벽을 낮추는 데 기여했다는 데 의의가 있다.

주제어 프로그래밍 환경, 유연성, 입문자, 맞춤형 교육, 파슨스 문제

ABSTRACT

Programming, a core component of computational thinking that enables individuals to understand a world increasingly shaped by software and artificial intelligence (SW·AI), continues to present high entry barriers for beginners. This study aims to develop a flexible programming environment that addresses the limitations of Parsons Problems while enhancing accessibility for novice learners. To provide personalized instruction tailored to learners' proficiency levels, the environment was designed to integrate block-based elements within text-based programming tasks. A 15-week course involving 85 university students was conducted to evaluate the usability of the proposed environment. The results revealed that students without prior experience in Python reported significantly higher levels of satisfaction in programming confidence, understanding of programming concepts, and perceived usefulness than those with prior experience. These findings suggest that the proposed environment effectively lowers the entry barriers for beginners in text-based programming and supports more adaptive, learner-centered instruction in computing education.

Keywords Programming Environment, Flexibility, Beginner, Personalized Education, Parsons Problems

†정회원 고려대학교 의과대학 의료빅데이터 연구소 연구교수
††종신회원 고려대학교 교육대학원 컴퓨터교육전공 부교수 (교신저자)

논문투고 2025년 10월 05일
심사완료 2025년 11월 05일
게재확정 2025년 11월 18일
발행일자 2026년 01월 30일

* 본 논문은 2025년 한국연구재단의 지원을 받아 수행된 연구임(RS-2025-00555855)

* 본 논문은 제1저자의 고려대학교 일반대학원 박사학위논문 일부를 발췌하여 요약, 정리한 것임.

1. 서론

SW·AI 기술의 빠른 발전으로 고급인재 양성 뿐 아니라 초중등 교육에서도 프로그래밍 교육의 중요성 및 필요성이 강화되고 있다[1, 2]. 구글이나 마이크로소프트 등과 같은 세계적인 IT 기업 뿐 아니라 영국, 핀란드, 일본 등 국가적 차원에서도 AI·SW 프로그래밍 교육은 국가의 경쟁력을 끌어올리는 기반으로 고려하고 있다[3]. 프로그래밍 교육의 중요성은 입문자 교육의 강화에도 영향을 주고 있다. 프로그래밍 입문자의 경우, 언어, 구문 뿐 아니라 표현에 대한 미숙으로 인해 프로그래밍 자체에 대한 흥미를 감소시키는 요인으로 작용하고 있다[4, 5].

교육의 관점에서 프로그래밍 교육을 실시할 때, 입문자의 요구가 충분히 반영되어야 하지만, 현재의 프로그래밍 교육에서는 내용에 대한 충분한 이해가 부족하기 때문에 입문자 스스로 무엇을 원하는지에 대한 내용을 판단하기 어렵다는 점에 주목할 필요가 있다. 즉, 관심과 능력이 부족한 상태에서 프로그래밍을 진행하거나, 관심이 있다 하더라도 표현할 능력이 부족한 의식적 무능력(Conscious Incompetence)의 상태가 지속되어 프로그래밍 자체를 포기할 가능성을 높이게 된다[6]. 입문자의 표현 능력을 높이기 위해서는 쉬운 이해를 기반으로 한 표현의 다양성을 높여 줄 수 있도록 하기 위해 교육용 프로그래밍 언어가 개발되었다.

2000년대 초에 등장한 앨리스는 객체지향 패러다임을 고려하여, 명령어 타일을 마우스로 조합하여 프로그램을 구성할 수 있도록 설계되었다[7]. 앨리스는 패러다임의 어려움으로 인하여 객체의 속성 설정, 많은 명령어와 속성 정보에 대한 이해로 인해 학습 부담을 증가시키게 되었다. 학습자의 접근성 향상을 위해 Scratch[8]를 시작으로, App Inventor[9], Snap![10], Code.org[11], 한국의 Entry[12] 등 블록 기반 프로그래밍 언어가 개발되었다[13]. 블록 기반 프로그래밍 언어는 입문자가 프로그래밍에 대한 개념을 쉽게 습득할 수 있도록 하는데 도움을 주었다[14].

블록 기반 프로그래밍은 쉬운 접근성은 텍스트 기반 언어로 전환시 학습의 연계가 어려운 것으로 보고되었다[15, 16]. CSTA 2016에서도 K-5 대상의 프로그래밍 학습 과정에서 블록 기반 언어는 프로그래밍 개념을 가르치는데 한계가 있다고 하였다[17]. 즉, 프로그래밍에 대한 개념 습득을 위해서는 블록 기반 언어의 익숙함을 텍스트 기반 언어로 전이할 필요가 있는 것으로 해석할 수 있다.

텍스트 기반 프로그래밍 언어의 어려움을 감소하기 위해 Code.org의 AppLab[18], Google의 Blockly[19], Pencil Code[14] 등의 환경이 개발되었다. 블록을 텍스트로, 텍스트를 블록으로 1:1 전환할 수 있는 환경이라 할지라도 번역과 유사한 형태로 학습 효과로의 기대는 제한적이었다.

번역과도 같은 환경의 극복을 위해 Parsons problems[20]는 파이썬의 명령어를 블록의 형태로 제공하여 입문자의 텍스트 기반 프로그래밍에 대한 진입장벽을 낮추는 데 기여하였다[21, 22]. Parsons problems는 제공된 명령어의 순서를 조합하는 형태로, 수정이나 명령어추가와 같은 유연성에 한계를 나타내었다.

입문자의 프로그래밍 학습 효과는 입문자 스스로 프로그래밍에 대한 능숙함의 단계를 향상시킬 수 있어야 하며, 환경을 통해 능숙함이 습득될 수 있을 것으로 판단된다[23]. 즉, 입문자 수준에서 프로그래밍 환경을 편리하게 구성하는 것, 프로그래밍 과정을 통해 프로그래밍에 대한 능숙함이 향상되도록 하는 것이다. 이에 본 연구는 입문자의 프로그래밍 능숙도를 향상시키는 데 기여할 수 있는 프로그래밍 환경을 개발하기 위한 목적으로 진행되었다.

2. 관련 연구

2.1 블록기반 프로그래밍 환경의 특징

2006년에 처음 소개된 Scratch는 지난 몇 년 동안 어린이들에게 프로그래밍을 가르치는데 가장 널리 사용되는 언어 중 하나가 되었다[24]. 스크래치는 사용 가능한 명령어를 시각적으로 제공해주며, 마우스로 명령어를 조합하여 프로그래밍 활동을 하기 때문에 입문자도 쉽게 프로그래밍 할 수 있다. 또한 자신이 만든 결과물을 전 세계 다른 학생들과 공유할 수 있으며 프로그램 코드도 열람할 수 있다. 그러나, 스크래치 플랫폼에서만 사용할 수 있기 때문에 C, Python, C++ 등 다른 범용적인 프로그래밍 언어와 함께 사용할 수 없고 사용할 수 있는 명령어가 제한되어 있어 현실 세계에서 사용할 수 있는 프로그램을 만드는데 한계가 있다[25].

(a) Pencil Code Blocks (b) Pencil Code Text

Figure 1. An example of the text/blocks transition

이러한 한계를 해결하기 위해 Figure 1과 같은 Pencil Code는 블록으로 조합한 명령어를 텍스트 명령어로 변환해주는 기능을 제공해준다[26]. 그러나, 텍스트 명령어를 보는 것만으로는 프로그래밍 언어를 학습했다고 보기 어렵다. 프로그래밍 언어를 학습하기 위해서는 학습자가 직접 명령어를 입력해보고 수정, 삭제, 추가해보는 활동을 통해 익히는 과정이 필요하지만, 기존의 프로그래밍 환경은 이러한 기능을 제공하지 않는다.

Table 1은 컴퓨터 교육 전문가 8인이 주요 블록 기반 프로그래밍 환경을 분석한 것이다. Blockly, Pencil Code, Parsons Problems는 텍스트 기반 프로그래밍 언어인 파이썬을 사용하고 있지만, 학습자가 별도의 입력, 수정, 삭제하는 과정을 제공하지 않는다. Code.org와 Parsons

Problems는 문제 풀이 형태를 제공하고 있으며, Scratch만 학습자가 문제를 등록할 수 있는 기능을 제공한다. 그리고, Code.org만 프로그래밍 학습 단계를 제공하는 것으로 나타났다.

Table 1. Analysis Results of Block-Based Programming Environments

구분	Scratch	Code.org	Blockly	Pencil Code	Parsons Problems
Block-based	O	O	O	O	O
Transfer (Using text-based programming language)	-	-	O	O	O
Transfer (Manipulating text-based programming language)	-	-	-	-	-
Problem solving and self-assessment	-	O	-	-	O
Problem registration	O	-	-	-	-
Provision of programming learning stages	-	O	-	-	-

2.2 Parsons Problems

Parsons Problems는 텍스트 기반 언어를 이해하기 위해 개발된 프로그래밍 퍼즐이다[22]. 기존 블록 기반 프로그래밍 환경과 다르게 파이썬 명령어를 사용하기 때문에 범용적인 프로그래밍 언어로 전이하는데 도움이 될 수 있다. 사용자가 명령어를 입력하는 방식이 아니기 때문에 파이썬 명령어를 모르는 입문자도 파이썬 코드가 작성되어 있는 코드 블록을 마우스로 드래그 앤 드롭하여 올바른 순서로 정렬할 수 있다. Figure 2는 Parsons Problems 웹사이트에서 제공하는 예시 문제를 나타낸 것이다. 왼쪽에 제시된 6개의 명령어를 올바른 순서대로 오른쪽 공간으로 이동시켜야 하며, 들여쓰기도 고려해야 한다. “Get feedback”을 눌러 명령어를 올바르게 배치했는지 확인할 수 있으며, 오류가 발생한 경우, 오류의 위치가 빨간색으로 표시된다.

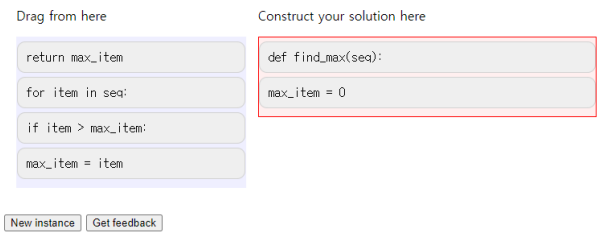


Figure 2. Parsons Problems

Parsons Problems는 다음과 같은 장점이 있다. 첫째, 문제 풀이 형태로 구성되어 있어 입문자가 흥미를 가지고 프로그래밍을 할 수 있으며 평가 결과를 스스로 확인할 수 있다. 둘째, 명령어를 블록 형태로 제공해줌으로써, 입문자가 파이썬 명령어를 타이핑하지 않고 문제해결 절차인 알고리즘에 집중할 수 있다. 셋째, 오류가 발생한 부분을 색깔로 표시하여 입문자가 오류 발생 원인을 빠르고 정확하게 인식하게 해준다. 그러나, 순서를 조합하는 과정만으로는 프로그래밍 학습이 이루어졌다고 보기 힘들다. 효과적인 프로그래밍 학습이 이루어지기 위해서는 사용자가 직접 코드를 입력하고 수정하고 삭제하고, 이 과정에서 발생하는 오류를 해결하는 과정을 통해 프로그래밍 학습이 이루어져야 한다.

3. 연구 방법

3.1 파이썬 프로그래밍 환경 설계 및 개발

개발된 프로그래밍 환경은 Figure 3과 같이 “프로그래밍 문제 등록”, “프로그래밍 활동”, “데이터베이스”로 구성되어 있다. “프로그래밍 문제”는 프로그래밍 언어로 구현해야 하는 프로그램에 대한 설명 및 문제 상황을 의미하며, 교수자가 학습자에게 문제를 제공할 때 사용할 수 있다. 프로그래밍 문제를 등록할 때는 문제에 대한 설명(Description) 및 구현할 때 사용할 수 있는 명령어를 제공할 수 있다. 등록된 프로그래밍 문제는 “Problems DB”에 저장되며, 학습자는 “Programming Workspace”에서 확인할 수 있다.

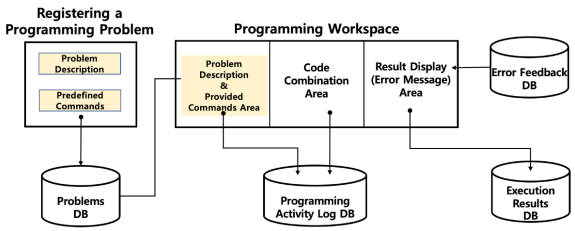


Figure 3. System Architecture

“Programming Workspace”는 “Problem Description & Provided Commands Area”, “Code Combination Area”, “Result Display(Error Message) Area”로 구성되어 있다. “Problem Description & Provided Commands Area”에서는 사용자가 프로그래밍 언어로 구현해야 하는 프로그램에 대한 설명 및 문제 상황을 확인할 수 있다. “Code Combination Area”에서는 문제 풀이에 사용할 수 있는 명령어를 확인할 수 있다. “Result Display(Error Message) Area”에서는 프로그램의 실행 결과 또는 오류 내용을 확인할 수 있다. “Programming Workspace”에 사용자가 명령어를 이동, 조합, 추가, 수정, 삭제하는 과정이 “Programming Activity Log DB”에 저장된다. 프로그램을 실행한 결과 또는 오류 내용은 “Exectution Results DB”에 저장되며, 오류가 발생한 경

우 “Error Feedback DB”를 통해 사용자가 문제 해결에 도움이 되는 피드백이 제공된다. 이와 같이 드래그 앤 드롭 기반의 직관적 인터페이스, 즉각적인 실행 피드백, 단계별 난이도 조절 기능을 반영하여 학습자가 자신의 수준에 맞게 프로그래밍 할 수 있는 유연성을 갖춘 프로그래밍 환경으로 설계하였다.

개발에 사용된 프로그래밍 언어는 HTML과 PHP 7.3, Javascript, Ajax이며, 데이터베이스는 mariadb-10.0을 사용하였다. 블록 기반의 명령어는 공개된 Nestable 2[27] 소스 코드, 파이썬 프로그래밍 환경은 공개된 Brython[28] 소스 코드를 사용하였으며, 파이썬의 버전은 3.9.0이다.

3.2 사용성 평가를 위한 분석 도구 및 문항 개발

개발된 프로그래밍 환경의 사용성 평가를 위해 기존 연구를 기반으로 Table 2와 같이 “사용 자신감, 프로그래밍 기능 이해, 유용성”으로 구성된 설문지를 개발하였다.

개발된 설문 항목은 프로그래밍 전문가 8인의 서면 검토와 1회의 예비 조사를 통해 수정 및 보완하였다. 설문은 수업 후 온라인으로 1회 진행하였으며 5점 리커트 척도를 사용하여 설문의 내용에 매우 동의하면 5점, 전혀 동의하지 않으면 1점이다.

Table 2. Factors and Survey Items

Factor	Survey Items
Confidence in use	Ease of using commands
	Confidence in using commands
	Confidence in using the programming environment
	Knowledge and skills in usage
	Confidence in creating desired outcomes
Understanding of programming commands	Understanding of [Output] commands
	Understanding of [Input] commands
	Understanding of [Operation] commands
	Understanding of [Condition] commands
	Understanding of [Loop] commands
	Understanding of [List] commands
	Understanding of [Function] commands
	Understanding of algorithms
Usefulness	Helpfulness in programming activities
	Willingness to spend more time using it
	Willingness to continue using it after class

3.3 연구 대상 및 수업 내용

수업은 자발적으로 신청한 대학생 85명을 대상으로 일주일에 3시간씩 15주 동안 진행되었다. 이 중 파이썬 경험이 있는 학생은 38명(경험자)이었으며, 파이썬 경험이 전혀 없는 학생(입문자)은 47명이었다.

Table 3. Weekly Class Content

Weeks	Contents
1	Basic usage methods (moving, modifying, adding, and deleting commands)
2	Usage of output commands (numbers, text, etc.)
3	Input methods using the keyboard
4	Calculations using arithmetic, comparison, and logical operations
5	Producing different outputs based on conditions (if, elif, else)
6	Usage of the for loop command
7	Usage of the while loop command
8	Midterm examination
9–10	List-related commands (adding, modifying, deleting, and displaying data)
11	Usage of list commands
12	Built-in functions provided by Python
13	Creating user-defined functions
14	Individual project
15	Final examination

Table 3은 15주간 진행된 수업 내용을 나타낸 표이다. 1주차에 학습자는 개발된 프로그래밍 환경에서 제공하는 기본 기능인 명령어 이동, 수정, 추가, 삭제하는 방법에 대해 예제 문제를 통해 학습하였다. 2주차에는 숫자 또는 문자가 포함된 출력 명령어 사용방법에 대해 학습했으며 3주차에는 키보드를 사용한 입력 방법에 대해 학습하였다. 4주차에는 사칙연산, 비교연산, 논리연산의 사용 방법에 대해 학습하였으며, 5주차에는 조건(if, elif, else)에 따라 다른 결과를 출력하는 방법에 대해 학습하였다. 6주차와 7주차에는 각각 반복 명령어 for와 while의 사용 방법에 대해 학습하였다. 8주차에는 중간고사가 진행되었다.

9주차 ~ 11주차에는 리스트를 만들고, 데이터를 추가, 수정, 삭제, 출력하는 방법에 대해 학습하였으며, 12주차에는 min, max, sum과 같이 파이썬에서 제공하는 기본 함수의 사용 방법에 대해 학습하였다. 13주차에는 def를 사용하여 함수를 만드는 방법에 대해 학습하였으며, 14주차에는 배운 내용을 기반으로 개별 프로젝트가 진행되었다. 15주차에는 기말고사가 진행되었다.

4. 연구 결과

4.1 파이썬 프로그래밍 환경 UI

Figure 4는 프로그래밍 문제 등록 화면이며 교수자가 문제를 등록할 때 사용한다. “Title”에 프로그래밍 문제의 제목을 작성할 수 있으며, “Public”과 “Close”를 통해 공개 여부를 선택할 수 있다. “Text”에는 사용자가 만들어야 하는 프로그램에 대한 설명을 작성할 수 있으며, “Hint”에는 프로그램을 만들기 위해 필요한 힌트를 작성할 수 있다. “Random”에서 “Random”에 체크를 하는 경우, 제

공된 명령어가 무작위로 섞여서 사용자에게 제공되지만, “Fixed”를 선택하면 문제를 등록할 때와 동일한 순서대로 사용자에게 명령어가 제공된다. “Modify”와 “Add”에서 “Allowed”를 선택하면 사용자가 명령어를 수정하거나 추가할 수 있지만 “Deny”를 선택하면 사용자는 제공된 명령어만 사용할 수밖에 없다. “Please enter the command”에 사용자에게 보여지는 파이썬 명령어를 입력할 수 있으며, “Please enter the comment”에는 명령어의 주석을 입력할 수 있다. “+”를 누르면 명령어 블록이 등록된다.

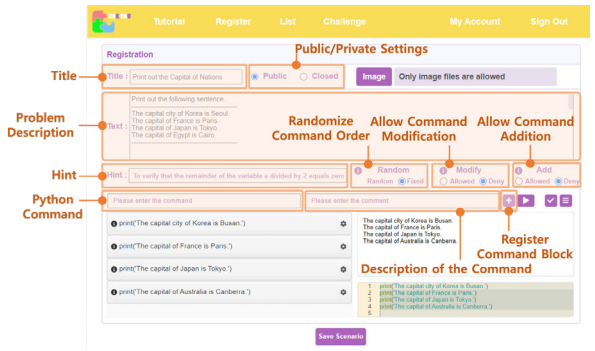


Figure 4. Registering a Programming Problem

Figure 5는 Programming Workspace를 나타낸 것이다. “Problem Description Area”에는 교수자가 등록한 프로그래밍 문제의 내용이 나타나며, 학습자는 이를 바탕으로 프로그램을 만들 수 있다. “Provided Commands Area”에는 교수자가 제공한 명령어가 나타나며, 사용자는 제공된 명령어를 사용하여 프로그램을 만들 수 있다. 만약, 교수자가 프로그래밍 문제 등록 화면에서 명령어를 무작위로 제공하도록 설정하였다면 명령어의 순서가 무작위로 제공되며, “Modify”와 “Add”가 가능하도록 설정하였다면, 사용자는 톱니바퀴 아이콘을 눌러 명령어를 추가하거나 수정할 수 있다. 사용자는 명령어를 “Code Combination Area”에 조합하여 프로그래밍 할 수 있으며, “Result Checking Area”에서 실행 결과를 확인하거나 오류 메시지를 확인할 수 있다. “Code Combinataion Area”에 블록 명령어를 조합하면 “Combined Text Code Area”에 텍스트 형태로 표시된다.

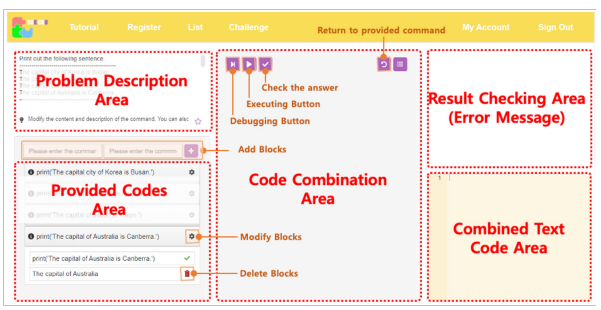


Figure 5. Programming Activity UI

Figure 6은 “Provided Commands Area”에서 “Code Combination Area”으로 명령어를 드래그 앤 드롭하여 프로그래밍하는 과정을 나타낸 것이다. 옮겨진 명령어가 위치할 곳에 옅은 노란색으로 표시된다. 명령어는 “Provided Commands Area”와 “Code Combination Area” 내에서도 이동할 수 있다

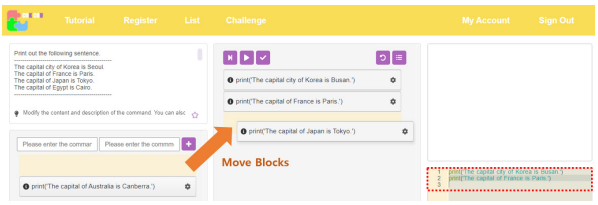


Figure 6. Move Blocks

Figure 7은 명령어를 수정하는 과정을 나타낸 것이다. 실행 버튼(▶)을 눌러 명령어 실행 결과를 “결과 확인 영역”에서 확인할 수 있다. 명령어의 수정이 필요하다고 판단되면, 톱니바퀴 모양의 아이콘을 눌러 스스로 수정할 수 있다.

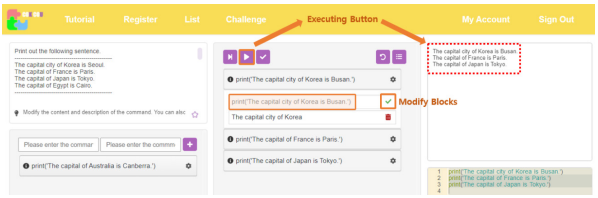


Figure 7. Executing and Modify Blocks

Figure 8은 사용자가 명령어를 추가하는 과정을 나타낸 것이다. ‘+’ 모양의 아이콘을 눌러 파이썬 명령어와 명령어에 대한 설명을 추가할 수 있다.

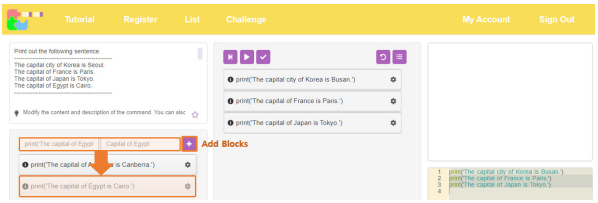


Figure 8. Add Blocks

Figure 9는 디버깅 과정을 나타낸 것이다. 디버깅 버튼(▶)을 ‘click’하여 컴파일된 결과를 확인할 수 있다. 컴파일 결과는 오른쪽 상단의 “결과 확인 영역”에서 확인할 수 있다.

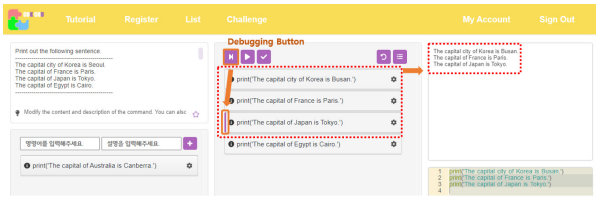


Figure 9. Execution by line for debugging

Figure 10은 디버깅 버튼(▶)을 'click'한 경우, 컴파일 과정에서 오류가 발생한 경우이다. 오류가 발생하면, "결과 확인 영역"에 오류 메시지가 출력된다. 실행 버튼(▶)을 'click' 할 경우에는 오류가 처음 발생한 지점이 노란색으로 표시된다.

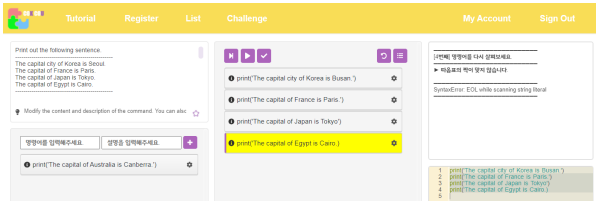


Figure 10. Error line is marked in yellow

Figure 11은 프로그래밍 요소의 문제 풀이 순서를 나타낸 것이다. 입문자는 "print, input, operator, if statement, list, for statement, function"으로 구성된 7개의 프로그래밍 요소에 대해 "따라하기", "배치하기", "수정하기", "추가하기" 형태로 학습할 수 있다.

"따라하기"는 Figure 5의 "제공된 명령어 영역"에 제시된 명령어를 "프로그래밍 활동 영역"으로 그대로 이동시키는 활동으로 가장 쉬운 유형이다. 입문자는 "따라하기" 유형을 통해 파이썬 명령어를 확인할 수 있다. "배치하기"는 무작위로 배치된 명령어를 올바른 순서대로 나열하는 것이며, 순서를 모르는 경우 "따라하기"에서 활동한 내용을 참고할 수 있다. "수정하기"는 "따라하기"와 동일한 순서로 명령어가 제공되지만, 명령어의 일부 내용이 누락되어 있어 입문자가 수정해야 한다. "추가하기"도 "따라하기"와 동일한 순서로 명령어가 제공되지만 명령어 한 줄이 누락되어 있어 입문자가 추가해야 한다.

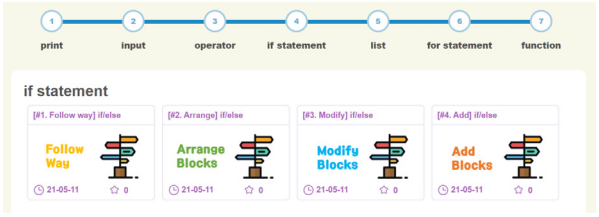


Figure 11. Problem-Solving Sequence (Based on Programming Elements)

Figure 12는 문제 풀이 결과를 나타낸 것이다. 정답 확인 버튼(✔)을 누르면, 정답과 얼마나 일치하는지 표시되며 정답과 완전히 일치하면 100%로 표시된다.

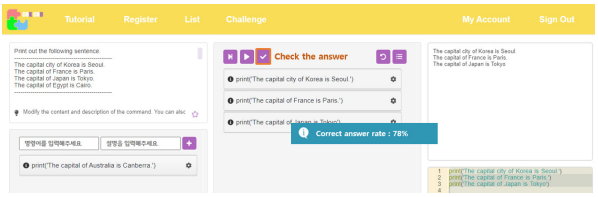


Figure 12. Problem-Solving Results (Accuracy Rate)

정답률을 계산하는 방식은 Levenshtein distance 알고리즘을 사용하였으며, 수식은 다음과 같다.

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j)=0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j)+1 \\ \mathrm{lev}_{a,b}(i,j-1)+1 \\ \mathrm{lev}_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

교사가 등록해 둔 정답과 학생이 제출한 명령어를 한 단어씩 비교하여 유사도를 계산한다.

4.2 사용성 평가 결과

본 연구에서 개발한 프로그래밍 환경이 입문자에게 얼마나 도움이 되었는지 확인하기 위해 다음과 같은 사용성 평가를 진행하였다. 평가 항목은 '전혀 동의하지 않는다(1점)' ~ '매우 동의한다(5점)'로 구성된 5점 리커트 척도를 사용하였다.

Table 4는 본 논문에서 개발한 프로그래밍 환경에 대한 두 집단의 사용성 평가 결과를 나타낸 것이다. 독립표본 t-검정을 통해 두 집단 간 평균 차이를 검증한 결과 모든 설문 항목에서 입문자가 경험자에 비해 높은 응답 결과를 나타내었으며, 특히, "명령어 사용 자신감"과 "입력/연산/조건 명령어의 이해"에 대해서는 유의수준 .05 수준에서 유의미한 차이가 있었다.

Table 4. Usability Evaluation Results

Factor	Survey Items	입문자 (N=47)	경험자 (N=38)	t
Confidence	Ease of using commands	4.53 (0.50)	4.29 (0.80)	1.622
	in use	4.30 (0.81)	3.84 (1.00)	2.327*
	Confidence in using the programming environment	4.34 (0.64)	4.24 (0.97)	0.567
	Knowledge and skills in usage	4.49 (0.55)	4.26 (0.76)	1.594
	Confidence in creating desired outcomes	4.32 (0.66)	4.32 (0.77)	0.022

Factor	Survey Items	입문자 (N=47)	경험자 (N=38)	t
Understanding of programming commands	Understanding of [Output] commands	4.55 (0.54)	4.24 (0.85)	1.985
	Understanding of [Input] commands	4.60 (0.50)	4.24 (0.85)	2.300*
	Understanding of [Operation] commands	4.62 (0.49)	4.29 (0.84)	2.137*
	Understanding of [Condition] commands	4.64 (0.49)	4.32 (0.81)	2.163*
	Understanding of [Loop] commands	4.60 (0.54)	4.29 (0.84)	1.956
	Understanding of [List] commands	4.55 (0.54)	4.29 (0.84)	1.679
	Understanding of [Function] commands	4.55 (0.54)	4.26 (0.86)	1.807
	Understanding of algorithms	4.55 (0.54)	4.29 (0.80)	1.730
Usefulness	Helpfulness in programming activities	4.66 (0.48)	4.16 (0.89)	3.139**
	Willingness to spend more time using it	4.15 (0.81)	3.47 (1.27)	2.849**
	Willingness to continue using it after class	4.11 (0.89)	3.39 (1.31)	2.864**

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

또한, “프로그래밍 활동에 도움”이나 “더 많은 시간 사용 의지”, “수업 후 계속적인 사용 의지”에 대한 설문 항목에 대해서 입문자가 경험자에 비해 유의수준 .01 수준에서 유의미한 차이가 나타났다. 프로그래밍 입문자의 경우, 프로그래밍 언어에 대한 사용의 어려움이 크게 나타남[29]을 고려할 때, 본 연구에서 개발한 프로그래밍 환경은 프로그래밍에 대한 입문 과정의 두려움을 낮추는데 기여한 것으로 판단된다.

“프로그래밍 내용 이해”를 구체적으로 살펴보면 출력에서 알고리즘 이해로 이어지는 모든 과정의 이해도가 유사하게 나타났다. 프로그래밍 위계가 높아질수록 입문자의 실패율이 높아진다는 연구[29]와 다른 결과이다. 즉, 개발된 프로그래밍 환경의 사용은 쉬우며, 프로그래밍에 대한 이해를 높이는데 기여한 것으로 해석할 수 있다.

5. 결론

프로그래밍 입문자가 외부의 도움없이 텍스트 기반 프로그래밍 활동을 하는 것은 쉽지 않다. 본 연구에서 개발한 프로그래밍 환경은 다음과 같은 기능을 제공하여 입문자가 스스로 텍스트 기반 프로그래밍 활동을 할 수 있는 경험을 제공하였다.

첫째, 블록 기반 프로그래밍 활동으로 텍스트 기반 언어에 대한 부담감을 줄일 수 있다. 텍스트 기반 프로그래밍 언어를 사용한 입문자는 명령어 사용의 어려움으로 프로그래밍에 대한 자신감이 떨어지는 문제가 있지만, 개발된 프로그래밍 환경은 사용자가 명령어를 입력하지 않고 마우스로 드래그 앤 드롭하는 환경을 제공함으로써 사용자가 자신감을 가지고 프로그래밍 할 수 있다.

둘째, 문제 풀이 중심의 활동을 통해 학습과 평가가 동시에 수행할 수 있도록 한다. 기존의 Parsons Problems는 코드의 수정이 불가능하지만, 본 연구에서 개발한 환경은 마우스를 이용하여 명령어를 이동하는 것뿐만 아니라 파이썬 명령어를 직접 수정하고 추가할 수 있도록 구성하였다. 이를 통해 학습자는 파이썬 명령어를 점진적으로 활용하고 이해하는 능력을 향상시킬 수 있다.

셋째, Parsons Problems와 달리 자신의 수준에 맞는 프로그래밍 활동을 선택해서 진행할 수 있는 맞춤형 학습이 가능한 환경을 제공한다. “따라하기, 배치하기, 수정하기, 추가하기” 기능을 이용하여 학습자가 자신의 수준에 맞는 학습을 선택하여 진행할 수 있는 학습의 단계를 제공한다.

넷째, 프로그래밍 학습의 순서를 제공한다. 프로그램을 개발하기 위해서는 다양한 프로그래밍 요소가 필요하다. 입문자가 수준에 맞게 프로그래밍을 단계별로 진행할 수 있도록 “출력, 입력, 연산, 조건, 리스트, 반복, 함수”의 순서를 제공한다.

본 연구에서 개발한 프로그래밍 환경에 대한 사용성 평가 결과, 입문자들이 쉽게 사용할 수 있으며, 프로그래밍 내용을 이해하는데 도움을 준 것으로 확인되었다. 입문자를 위한 프로그래밍 교육을 위해 다음과 같은 논의가 필요하다.

첫째, 입문자가 텍스트 기반 프로그래밍 언어를 학습하는데 도움을 주는 환경 개발에 대한 후속 연구가 필요하다. 블록 기반 프로그래밍 활동이 곧 텍스트 기반 프로그래밍 활동으로 연결되는 것은 아니다[14]. 텍스트 기반 프로그래밍을 바로 진행하는데는 블록 명령어를 옮기는 것과 다른 인지 부담이 발생한다[30]. 따라서 향후 연구로, 텍스트 기반 프로그래밍 언어를 제공하고 해당 언어를 기반으로 직접 타이핑하면서 프로그래밍 할 수 있는 환경을 제공해 준다면, 블록 기반 언어에서 텍스트 기반 언어로의 전이가 더욱 용이할 것으로 판단된다.

둘째, 본 연구에서 제안한 프로그래밍 환경의 수정, 추가, 삭제 기능을 활용하여 학습자 수준에 따른 다양한 콘텐츠가 개발 및 제공될 필요가 있다. 외국어를 배우는 과정과 비슷하게 프로그래밍 언어를 습득하는 과정에서 명령어를 직접 추가하고 수정 및 삭제하는 과정을 통해 학습자가 스스로 프로그래밍 언어를 습득하는 과정을 경험하도록 할 필요가 있다.

셋째, 최근 인공지능 기술을 활용하여 사용자의 오류 내용을 분석하고 해결하기 위한 피드백을 실시간으로 제공해주는 기능이 추가될 필요가 있다. 교사 한 명이 여러 명의 학생을 돌볼 수 없기 때문에 학습자 스스로 프로그래밍을 할 수 있는 환경을 제공할 필요가 있다[31].

프로그래밍 학습 과정에서 학습자의 자기 효능감이나 동기 부여는 학습의 성공에 많은 영향을 미친다[32, 33]. 따라서 본 연구에서 개발한 프로그래밍 환경이 프로그래밍 기능을 이해하는데 도움을 주는 것 외에도 프로그래밍에 대한 관심과 인식이 높아졌는지, 학습 동기부여에 얼마나 영향을 주었는지를 확인할 수 있는 후속 연구가 진행될 필요가 있다. 개발된 프로그래밍 환경이 입문자의 프로그래밍 과정에 얼마나 기여했는지에 대한 평가도 고려한다면, 개발된 프로그래밍 환경의 교육적 가치가 좀 더 분명히 검증될 수 있을 것이다. 또한, 학교 현장에서 활용되기 위해서는 논의 및 제한 사항을 고려하여 더 많은 학습자를 대상으로 적용이 될 필요가 있다.

참고문헌

[1] Norovich, T. (2022). The importance of the formation of web programming competencies in schoolchildren. *European Journal of Research and Reflection in Educational Sciences, 10*(1).

[2] Furuta, T., Okugi, Y., & Knezek, G. (2023). Teaching Coding and Computational Thinking with Model Train Robotics: Social Factors That Motivate Students to Learn Programming. *In Teaching Coding in K-12 Schools: Research and Application.*

[3] Hainey, T., Baxter, G., & Ford, A. (2020). An evaluation of the introduction of games-based construction learning in upper primary education using a developed game codification scheme for scratch. *Journal of Applied Research in Higher Education, 12*(3), 377-402.

[4] Kim, J., Kim Y. (2019). A Study on the Utility of Online Learning for Expression Education in SW Education. *Korean Journal of Converging Humanities, 7*(2), 1-15.

[5] Lee, D., Kim, Y., & Lee, Y. (2025). Development and Application of a Framework for Analyzing Cognitive Competence Demand Levels in Programming Learning Tasks. *The Journal of Korean Association of Computer Educatio, 28*(3). https://doi.org/10.32431/kace.2025.28.3.004

[6] Howell, W. (1981). *Empathic communication*. Waveland Press.

[7] Sykes, E. (2007). Determining the effectiveness of the 3D Alice programming environment at the computer science I level. *Journal of Educational Computing Research, 36*(2), 223-244.

[8] Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(4), 1-15.

[9] Turbak, F., Sherman, M., Martin, F., Wolber, D., & Pokress, S. (2014). Events-first programming in APP inventor. *Journal of Computing Sciences in Colleges, 29*(6), 81-89.

[10] Newley, A., Deniz, H., Kaya, E., & Yesilyurt, E. (2016). Engaging elementary and middle school students in robotics through hummingbird kit with Snap! visual programming language. *Journal of Learning and Teaching in Digital Age, 1*(2), 20-26.

[11] Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior, 52*, 200-210.

[12] Gim, N. G. (2021). Development of life skills program for primary school students: Focus on entry programming. *Computers, 10*(5), 56.

[13] Weintrop, D., & Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education (TOCE), 18*(1), 1-25.

[14] Weintrop, D., & Wilensky, U. (2019). Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms. *Computers & Education, 142*, 103646.

[15] Noone, M., & Mooney, A. (2018). Visual and textual programming languages: a systematic review of the literature. *Journal of Computers in Education, 5*(2), 149-174.

[16] Puente, E. L. (2022). Effect of the use of block-based languages in programming learning. In 2022 International Symposium on Computers in Education (SIIE). IEEE, 1-6.

[17] K-12 Computer Science Framework Steering Committee. (2016). *K-12 computer science framework*. ACM.

[18] Steinglass, A., Franke, B., & Filman, S. (2017). App Lab: A Powerful JavaScript IDE for Rapid Prototyping of Small Data-backed Web Applications. *In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,* 641-642.

[19] Broll, B., Lédeczi, A., Volgyesi, P., Sallai, J., Maroti, M., Carrillo, A., & Lu, M. (2017). A visual programming environment for learning distributed programming. *In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,* 81-86.

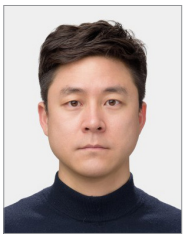
[20] Ericson, B., Foley, J., & Rick, J. (2018). Evaluating the efficiency and effectiveness of adaptive parsons problems. I*n Proceedings of the 2018 ACM Conference on International Computing Education Research*, 60-68.

[21] Shah, M. (2020). Exploring the use of Parsons problems for learning a new programming language. *Technical Report EECS-2020--88. Electrical Engineering and Computer Sciences*, University of California at Berkeley.

[22] Parsons, D., & Haden, P. (2006). Parson's programming puzzles: a fun and effective learning tool for first programming courses. *In Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*, 157-163.

[23] Vihavainen, A., Paksula, M., & Luukkainen, M. (2011). Extreme apprenticeship method in teaching programming for beginners. *In Proceedings of the 42nd ACM technical symposium on Computer science education* 93-98.

[24] Krafft, M., Fraser, G., & Walkinshaw, N. (2020). Motivating Adult Learners by Introducing Programming Concepts with Scratch. *In Proceedings of the 4th European Conference on Software Engineering Education*, 22-26.
[25] Mladenović, M., Boljat, I., & Žanko, Ž. (2018). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies, 23*(4), 1483-1500.
[26] Bau, D. (2015). Droplet, a blocks-based editor for text code. *Journal of Computing Sciences in Colleges, 30*(6), 138-144.
[27] Bushel, D. *Nestable2*. Retrieved October 5, 2025, from https://ramonsmit.github.io/Nestable2/
[28] Brython-dev. *Brython (Browser Python)*. Retrieved October 5, 2025, from https://github.com/brython-dev/brython
[29] Jenkins, T. (2002). On the Difficulty of Learning to Program, *Proceedings for the 3rd Annual conference of the LTSN Centre for Information and Computer Sciences.*
[30] Weintrop, D. (2019). Block-based programming in computer science education. *Communications of the ACM, 62*(8), 22-25.
[31] Park, H., Kim, J., & Lee, W. (2022). Development of CNN-Based Data Crawler to Support Learning Block Programming. *Mathematics, 10*(13), 2223.
[32] Tsai, C. Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior, 95*, 224-232.
[33] Theobald, M. (2021). Self-regulated learning training programs enhance university students' academic performance, self-regulated learning strategies, and motivation: *A meta-analysis. Contemporary Educational Psychology, 66*, 101976.



김용천

· 2004년 고려대학교 컴퓨터교육과(이학사)
· 2004년 고려대학교 컴퓨터교육학과(이학석사)
· 2024년 고려대학교 컴퓨터교육학과(이학박사)
· 2025년~현재 고려대학교 의과대학 의료빅데이터 연구소 연구교수

관심분야 : 정보교육, 프로그래밍 교육, 알고리즘
kimyc3766@gmail.com

김자미

· 1992년 이화여자대학교 교육학과(문학사)
· 1995년 이화여자대학교 교육학과(문학석사)
· 2011년 고려대학교 컴퓨터교육학과(이학박사)
· 2011년~2015년 고려대학교 컴퓨터학과 연구교수
· 2015년~현재 고려대학교 교육대학원 컴퓨터교육 전공 부교수

관심분야 : 정보교육, 교육과정평가, 에듀테크
celine@korea.ac.kr