JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 26, No. 4, pp. 1059-1068, Apr. 2025

Check for updates

컴퓨터 비전공자를 위한 예제 템플릿과 코드의 재구성 학습 기반 코딩 프로젝트 수행 방법에 대한 연구

정 혜 욱*
경기대학교 자유교양대학 교양학부 부교수

A Study on How to Conduct Coding Projects Based on Learning Example Templates and Code Reconstruction for Non-Computer Majors

Hye-Wuk Jung*
Associate Professor, Department of Liberal Arts, Kyonggi University, Suwon 16227, Korea

[요 약]

최근 AI 기술의 발달과 함께 소프트웨어의 수요와 중요성이 커지고 있다. 이에 많은 대학에서 최신 기술변화에 적응할 수 있는 인재 양성을 위해 전교생을 대상으로 코딩교육을 진행하고 있다. 학습자는 프로그래밍 언어의 문법을 배우며 연습문제 작성을 통해 프로그래밍의 원리와 다양한 기능을 익히고, 학습한 내용을 종합적으로 적용하여 코딩 프로젝트를 수행한다. 이러한 과정에서 프로그램을 처음 학습하거나 코딩 프로젝트 경험이 없는 컴퓨터 비전공자는 많은 어려움을 느끼게 되기 때문에, 학습자들이 보다 쉽고 능률적으로 프로젝트를 수행할 수 있는 효율적인 방법이 필요하다. 이에 본 연구는 컴퓨터 비전공자들이 효율적으로 코딩 프로젝트를 작성할 수 있도록 예제 템플릿과 코드의 재구성 학습 방법을 기반으로 프로그래밍 학습모델을 설계하고, 실제 수업에 적용하였다. 그리고 학습자의 예제 템플릿 유형 선택에 대한 변화, 프로젝트에서 사용한 파이썬 기능, 학습자 의견을 분석한 결과, 제안한 방법이 컴퓨터 비전공자에게 효율적으로 적용될 수 있음을 확인하였다.

[Abstract]

With the development of artificial intelligence (AI) technology, many universities are including coding education to cultivate talents in students to help them adapt to the latest technological changes. Learners learn programming grammar, the principles and various functions of programming, by solving practice problems, whereby they apply what they have learned to coding projects. In this process, non-computer majors who are learning programming for the first time or have no project experience may encounter difficulties. Therefore, they need an efficient method to perform projects more easily and efficiently. Accordingly, in this study a programming learning model was designed based on example templates and code restructuring learning methods for them to efficiently perform the projects and apply it to actual classes. By analyzing the changes in learners' choice of example template types, the Python functions used in the projects, and administering a learners' survey, the effectiveness of the proposed method for non-computer majors was confirmed.

색인어 : 컴퓨터 비전공자, 파이썬 프로그래밍, 코딩 프로젝트, 소프트웨어 교육, 프로그래밍 학습 모델
Keyword : Non-Computer Major, Python Programming, Coding Project, Software Education, Programming Learning Model

http://dx.doi.org/10.9728/dcs.2025.26.4.1059

Received 17 February 2025; **Revised** 07 March 2025
Accepted 13 March 2025

***Corresponding Author; Hye-Wuk Jung**

Tel: +82-31-249-1467
E-mail: wukj@kyonggi.ac.kr

Ⅰ. 서 론

최근 AI(Artificial Intelligence) 기술의 발달과 더불어 컴퓨터를 이용한 응용 기술이 다양하게 발달하면서 소프트웨어(Software)의 수요와 그 중요성이 증가하고 있다. IT(Information Technology) 변화에 적응할 수 있는 인재 양성을 위해 대학에서는 학습자들이 컴퓨터 프로그래밍(Computer Programming) 능력을 키울 수 있도록 필수적으로 코딩(Coding) 교육을 하고 있다[1]. 코딩 학습은 일반적으로 컴퓨터 언어에 대한 기본적인 문법을 익히고, 주어진 문제에 대한 결과를 얻기 위해 프로그램을 작성하는 과정으로 진행되며, 과제나 프로젝트(Project) 작업을 통한 학습 방법도 병행하게 된다.

코딩 과제는 단원별 연습문제 또는 응용문제에 대한 코드를 작성하는 내용으로 주어지며, 학습자는 작성한 코드 내의 오류를 수정하며 정답이 출력될 수 있게 프로그램을 완성한다. 이러한 과제는 학습자들이 프로그램의 문법별 기능을 익히는 데 도움이 된다[2]. 코딩 프로젝트는 프로그램의 문법 기능을 구체화하고, 학습한 내용을 복합적으로 적용해 보는 방법이다. 프로젝트를 수행하기 위한 계획단계에서는 아이디어를 도출하고, 프로그램의 전반적인 구조를 설계한다. 코드 작성 과정에서는 충분한 테스트를 진행하면서 발생하는 오류를 수정하고, 소스 코드의 내용을 보완하여 프로젝트를 완성한다[3]. 이러한 코딩 프로젝트 작업은 학습자가 한 학기 동안 배운 기능을 활용하여 아이디어를 도출하고 창의적인 결과물을 완성할 수 있으므로 과제나 시험과 함께 학습자에 대한 평가 방법으로 사용된다[4]. 또한 PBL(Project Based Learning)과 같이 학습자가 스스로 질문을 생성하고 관련 지식과 경험을 바탕으로 최종 결과물을 만들어 내는 자기 주도적인 학습 방법을 수업 전반에 걸쳐 적용한다[5]. 이 방법은 학습자가 프로젝트 수행 기간 설계, 문제해결, 의사결정과 같은 작업 내용에 대해 스스로 연구하고 새로운 정보를 창출한다[6]. 또한 교수와 구성원들과의 상호작용을 통해 프로젝트 학습을 계획하고 실천하며 결과를 도출하는 학습 방법이다[7]. 이와 같은 프로젝트 기반 학습은 여러 학문 분야에 응용되고 있으며, 컴퓨터 분야에서도 효과적인 프로그래밍 학습을 위한 교육 방법으로 적용되고 있다.

코딩 수업에서 학습자들이 경험하게 되는 프로젝트 활동은 창의력과 문제해결 능력을 키우고, 학습에 대한 동기부여와 코딩 실력 향상에 도움이 된다. 하지만 프로그래밍을 처음 배우는 초보자는 프로젝트를 수행하는 데 필요한 기초지식의 부족으로 인해 자기 주도적 학습 진행에 어려움이 있다. 예를 들어 프로젝트 제작 경험이 없는 경우 문제 계획, 알고리즘 설계, 기능별 구현, 오류수정 및 테스트와 같은 코딩 프로젝트 수행 방법에 익숙하지 않기 때문에, 작업 능률이 떨어지고 결과물의 완성도가 낮아질 수 있다. 또한 일반적으로 한 학기 동안 진행되는 수업을 통해 프로그래밍에 대한 지식을 익히며 코딩 프로젝트를 수행하게 되는데, 난이도가 높은 프로그램 문법의 내용을 이해하고 응용해서 프로젝트에 활용하기에는 시간이 충분하지 않아서 학습에 어려움을 느끼게 된다. 이러한 문제점을 해결하기 위해 동영상 강의를 이용한 보조적인 학습 방법을 병행하거나, 교수자와의 상호작용을 통한 피드백 전략의 연구 내용이 제시되었다. 그러나 대학에서 컴퓨터 비전공자가 파이썬(Python)과 같은 텍스트 기반 프로그래밍 언어를 이용하여 보다 쉽고 능률적으로 코딩 프로젝트를 수행할 수 있는 효율적인 학습 방법에 관한 연구가 부족하다. 또한 학습자 스스로 계획한 문제를 해결하는 과정에서 코딩 프로젝트 작성에 필요한 프로그램의 문법 기능을 선택 및 적용하고 다양한 문법을 조합하여 결과물을 완성하는 것은 매우 중요하다.

이에 본 연구에서는 컴퓨터 비전공자들의 코딩 프로젝트 작업에 대한 어려움을 최소화하고, 한 학기 동안 학습한 프로그래밍 기법을 최대한 활용하여 코딩 프로젝트를 수행할 수 있는 예제 템플릿과 코드의 재구성 학습 기반 코딩 학습모델을 설계하였다. 그리고 실제 수업에 적용 후 학습자의 예제 템플릿 유형 선택에 대한 변화와 프로젝트에서 사용한 파이썬 기능 및 학습자 의견을 분석하였으며, 제안한 학습모델을 향후 컴퓨터를 전공하지 않거나 코딩에 익숙하지 않은 학습자들에게 적용했을 때의 기대효과를 탐색해 보고자 한다.

Ⅱ. 코딩 프로젝트 관련 연구

현재까지 많은 연구자에 의해 프로젝트 학습을 적용한 다양한 프로그래밍 교육 방법들이 개발되었으며, 관련 연구들은 다음과 같다.

김시정 외 1명은 사전학습(Pre-Class)에서 프로그래밍의 핵심 문법과 수업 내용의 결과물을 배포하여 학습자들이 전체 알고리즘을 이해하고 프로젝트를 진행할 수 있게 하였다. 수업 시간(In-Class)에는 팀별 토의를 통해 프로젝트를 디자인하고, 제안된 내용에 대해 교수자로부터 피드백을 받는다. 이때, 교수자는 학습자에게 미니강의와 자료를 제공한다. 학습자는 교수자에게 최종 평가를 받기 위해 주차별 개인 과제와 팀별 연구 노트 작성을 수업이 종료할 때까지 사후 학습(Post-Class)에서 진행한다. 이러한 수업을 R 프로그램을 이용하여 인문계와 이공계 대학생을 대상으로 진행한 후 설문조사를 한 결과, 학습 내용에 대해 흥미롭고 긍정적인 반응을 보였지만, 일부 학습자들은 프로젝트 수행 작업이 부담스럽다는 의견을 보였다[8].

남충모 외 1명은 초등학생을 대상으로 하는 프로그래밍 수업에서 학습자에게 동기를 부여할 수 있도록 설계된 ARCS(Attention, Relevance, Confidence, and Satisfaction) 모델을 적용하여 코딩 프로젝트를 수행하게 하였다. 엔트리와 파이썬 프로그램을 이용하여 13주 동안 단계

별 교육을 실시한 후 학습자의 컴퓨팅 사고력 변화를 측정한 결과, 학생들은 전반적으로 프로그래밍에 대해 자신감과 흥미를 가지게 되었지만, 영어 사용이나 오류수정에 대해 어려움을 느끼는 것으로 나타났다[9].

구덕회는 스크래치(Scratch) 프로그램의 기능을 익히고, 프로젝트를 제작하는 과정에 디지털 스토리텔링(Digital Storytelling) 요소를 적용하는 프로그래밍 교육 방법을 제안하였다. 학습자는 디지털 스토리텔링의 개념을 이해하고 프로그래밍의 기본 기능에 대해 학습한 후, 프로젝트 주제 설정부터 스토리보드 제작까지의 전반적인 프로젝트 작업을 자기 주도적으로 진행하며, 결과물에 대한 발표와 평가 과정을 거쳐 디지털 스토리텔링 프로그램을 정리하는 과정으로 학습 방법을 설계하였다. 또한 학습자들이 흥미를 가지고 학습할 수 있도록 프로그램 학습을 위한 교재의 구성 방안을 제시하였다[10]. 그러나 이 방법을 초등학생 이외에 다양한 연령층의 학습자들을 대상으로 실제적인 수업에 적용해 볼 필요가 있다.

하석운 외 1명은 항공 우주 및 소프트웨어 공학을 전공하는 학습자들의 모바일 프로그래밍 능력을 향상시킬 수 있는 프로젝트와 연계된 교육 커리큘럼을 제안하였다. 총 15주의 학습 기간 중 첫 주부터 8주까지는 모바일 프로그래밍의 기본적인 내용에 대한 학습과 API(Applications Programming Interface)를 활용한 실습을 진행하고, 나머지 7주 동안에는 프로젝트 구현, 토의, 보고서작성 작업을 진행하도록 구성하였다. 학습자들의 최종 결과물을 통해 모바일 프로그래밍 교육에 효과가 확인되었고, 만족도와 호응도도 높게 나타났다[11]. 이와 같은 프로그램 개발을 중심으로 진행되는 방법은 초보 학습자에게 적용하기에는 어려움이 있다.

Moran Tsur 외 1명은 스크래치 프로그램을 단순화한 입문용 코딩 환경인 스크래치 마이크로월드(Scratch Microworlds)를 개발하였다. 학습자는 프로젝트 기반 접근 방식을 사용하여 다양한 실험을 하고 프로젝트를 설계하는 경험을 할 수 있게 하였다. 교수자는 학습자와 마이크로월드 프로토타입(Microworlds prototypes)의 상호작용을 관찰하고, 결과물에 대한 문서화와 피드백을 한다. 워크숍을 통해 약 90명의 교수자와 100명의 어린이 학습자와의 상호작용과 프로젝트를 진행하였다. 설문 조사와 토론 및 피드백의 내용을 수집 및 관찰한 결과, 학습자는 블록, 이미지, 사운드를 다양한 방식으로 결합하였고, 서로 다른 방식으로 코딩 블록을 사용하며 프로젝트를 완성하였다. 교수자는 단순화된 마이크로월드 프로토타입을 사용할 때 표현의 제약이 있으므로 좀 더 많은 도구가 필요하다는 의견을 나타냈다[12].

Mark Frydenberg 외 1명은 파이썬 코딩 프로젝트 작성 방법으로 PBL 접근 방식을 적용하여 설계하였다. 입문 과정인 CS(Computer Information Systems) 299에서는 기본적인 파이썬의 개념을 학습하며, 소그룹으로 나누어 코딩 연습을 수행하고, 주어진 문제에 대한 해결을 위해 솔루션을 구축한다. 고급 프로그래밍 과정인 ISA(Information Systems and Analytics) 330에서는 데이터 과학 응용 프로그램을 개발하기 위해 라이브러리를 다루는 프로그래밍 기술을 익힌다. 학습자들은 프로젝트 활동을 통해 코딩 기술을 익히며, 프로그램 개발 및 시연과 다른 사람들과의 작업을 공유하고 검토하는 과정을 경험한다. CS 299과 ISA 330을 수행한 학습자들은 창의적인 프로젝트를 합리적인 기대치 내에서 수행하였다. 학습자들은 프로젝트 발표 후 시행한 설문 조사에서 수학적 분석을 적용하는 것이 어려웠고, 데이터 분석기술을 이용하는데 더 많은 예제가 필요하다고 했다[13].

Akinode John Lekan 외 1명은 오프라인과 온라인에서 코딩 워크숍 프로그램을 진행하여 학습자들의 코딩 활동과 컴퓨팅 사고력의 변화를 관찰하였다. 초등학교 학생 50명은 오프라인 수업에서 기본적인 코딩 지식을 습득하고, 온라인 학습 플랫폼을 사용한 언플러그드(Unplugged) 활동 및 스크래치 프로그래밍을 통해 코딩 학습을 하였다. 학습자들은 스크래치 프로그램을 이용하여 게임, 애니메이션, 스토리를 디자인하고, 수학적 문제를 해결하였고, 퀴즈와 프로젝트로 평가를 받았다. 온라인과 오프라인 수업에 참여한 학습자들은 모두 컴퓨팅 사고력이 크게 향상된 결과를 나타냈고, 습득한 기술과 지식의 수준에 있어 차이가 없었다. 그러나 참가자들을 그룹으로 나누는 어려움과 인터넷 정전과 같이 기술적인 오류의 문제가 있었다[14].

김태령 외 1명은 블록 코딩 방법을 선행 학습한 학습자들에게 적용할 수 있는 파이썬의 교육 프로그램을 제안하였다. 학습자들은 파이썬 문법의 모방과 시연을 통해 코딩을 연습하고, 문법 적용 과정을 통해 자기 주도적 학습을 하며, 문제 해결 과정에서는 프로젝트 형태로 설계 및 개발을 진행한다. 이처럼 정해진 교육과정을 통해 전문가의 검토를 토대로 총 16차시의 교재를 개발하고, 예제 프로그래밍의 답안과 해설을 추가하였으며, 단계별 교재 내용에 대한 교수 방법과 전략을 제시하였다[15]. 하지만 제안한 프로그램을 실제 수업에 적용하고 효과를 검증할 필요가 있다.

이러한 연구들은 각 연구의 대상자에게는 효과적일 수 있으나, 컴퓨터 비전공자인 초보 학습자에게 적용하기에는 한계가 있다. 따라서 서론에서 언급하였듯이 본 논문에서는 컴퓨터 비전공자인 초보 학습자들이 코딩 프로젝트 작업에 대한 어려움을 최소화하고, 학습한 프로그래밍 기법을 최대한 활용하여 코딩 프로젝트를 수행할 수 있는 학습모델을 제안한다.

Ⅲ. 제안한 학습모델의 연구방법

3-1 코딩 학습모델 설계

본 연구에서는 대학생들이 한 학기 동안 프로그래밍에 대한 기본 지식을 익히고, 중간 과제와 기말 프로젝트를 수행하는 학습모델을 그림 1과 같이 설계하였다. 프로그래밍에 대한

문법을 학습하는 과정에서는 문법의 내용을 기본 기능과 확장 기능으로 분류하고, 이를 기반으로 예제 템플릿의 유형을 정의하였다. 예제 템플릿은 중간고사와 기말 프로젝트에 연계성을 갖도록 설계하였고, 이 연계성과 정의된 템플릿 유형을 바탕으로 학습자가 학습한 문법을 종합적으로 적용할 수 있는 중간고사와 기말 프로젝트 예제 템플릿으로 확장하였다.

제안한 학습모델을 실제 수업에 적용하기 위해 프로그래밍 언어로 대학생의 코딩교육에 많이 사용되는 파이썬을 사용하였다.

수업의 초반부터 중반까지의 학습 범위는 초보 학습자들이 기본적으로 배우는 파이썬 문법의 입력과 출력, 변수, 문자열, 수식, 조건문에 대한 기본 기능을 다룬다[16],[17]. 수업 중반 이후부터 후반까지는 중첩구조, 리스트, 반복문, 함수, 모듈에 해당하는 확장 기능에 대해 학습한다.

학습자는 프로그래밍 언어의 문법에 대한 이론적인 내용을 학습 후, 실습 시간에는 주어진 연습 예제와 응용문제를 작성하며, 해당 문법에 대한 내용을 익히고, 교수자가 제공한 예제 템플릿을 이용하여 제시한 조건에 맞게 코드의 재구성 연습을 한다.

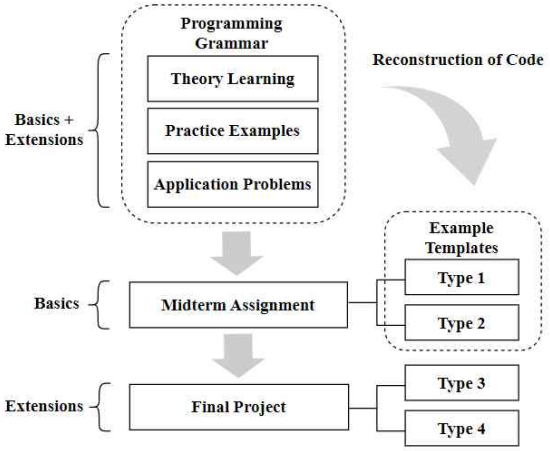

그림 1. 예제 템플릿과 코드의 재구성을 통한 코딩 학습모델
Fig. 1. Coding learning model through example templates and code restructuring

중간 과제와 기말 프로젝트에서 교수자가 제시하는 문제의 유형은 표 1과 같이 Type 1에서부터 Type 4로 분류되며, Type 1부터 Type 4의 유형별로 파이썬 문법 각 항목의 사용 여부를 O, X로 나타내었다.

중간 과제에 해당하는 Type 1과 Type 2는 학기 중반에 시행되므로 확장 기능 중 중첩구조를 조건문에서만 사용한다. 기말 프로젝트에서 사용되는 Type 3과 Type 4는 파이썬의 기본 기능과 확장 기능에 해당하는 모든 문법을 포함한다. 단, Type 4는 예제 템플릿을 사용하지 않으므로 학습자 스스로 코드의 구조를 자유롭게 작성한다.

그림 2는 예제 템플릿을 사용하는 Type 1, Type 2, Type 3, Type 4의 레이아웃을 나타내며, Type 1에서 Type 4로 갈수록 파이썬 문법의 기능이 확장되는 것을 볼 수 있다. 각 유형의 난이도와 복잡도는 Type 1에서 Type 4로 갈수록 높아진다.

중간 과제에서는 교수자가 제시한 예제 템플릿 Type 1과 Type 2를 이용하여 작성하고, 실행해 본 후 문제의 조건에 따라 프로그램의 주제를 계획하여 예제 템플릿 내용을 수정하여 콘텐츠를 변경하고, 학습한 파이썬 문법을 이용하여 코드의 기능을 추가한다. 이때 적용되는 파이썬 문법은 조건문에 해당하는 기본 기능까지 포함하며, 학습자는 각 한 개씩 제공되는 Type 1과 Type 2의 예제 템플릿 중 하나의 유형을 선택하여 작성할 수 있다.

표 1. 파이썬 문법 및 유형 분류
Table 1. Python grammar and type classification

Python Grammar		Type			
		Type 1	Type 2	Type 3	Type 4
Basic functions	Input/output	O	O	O	O
	Variables	O	O	O	O
	Strings	O	O	O	O
	Python formula	O	O	O	O
	Conditional Statements	O	O	O	O
Extensions	Nested-conditional statements	X	O	O	O
	Lists	X	X	O	O
	Loops	X	X	O	O
	Functions	X	X	O	O
	Modules	X	X	O	O

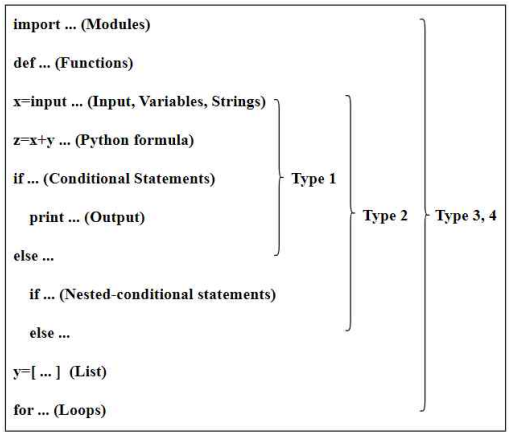

그림 2. 유형별 예제 템플릿의 레이아웃
Fig. 2. Layout of example templates by type

그림 3은 Type 1의 예제 템플릿 소스 코드와 실행 화면을 보여준다. 학습자는 ①에 해당되는 부분에서 콘텐츠를 변경하고, ②에서는 변경된 콘텐츠에 맞게 입력과 출력문, 수식, 조

건문 등 파이썬의 기본 기능을 이용하여 Type 1 프로그램의 기능을 확장하여 재구성한다.

기말 프로젝트에서는 문제해결을 위한 계획에서부터 코딩까지 학습자 스스로 진행하며, 기본 기능에 확장된 파이썬 기능을 모두 활용하여 프로젝트를 완성한다. 예제 템플릿으로는 Type 3과 Type 4 중 한 가지를 선택할 수 있으며, Type 3은 총 12개의 예제 템플릿을 제공하고 Type 4는 학습자가 새롭게 응용 프로그램을 설계하고 소스 코드를 작성한다.

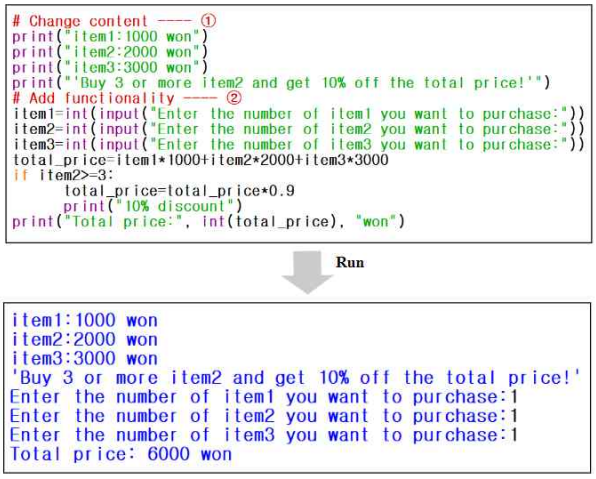

그림 3. Type 1의 예제 템플릿
Fig. 3. Example template for Type 1

3-2 연구 대상 및 평가 방법

본 연구는 경기도에 소재한 K 대학에서 2023년 1학기와 2학기에 개설된 파이썬 언어를 기반으로 학습하는 '소프트웨어기초', '컴퓨팅사고' 과목의 수강자 355명을 대상으로 하였다. 수업 형태는 하이브리드(비대면 + 대면)로, 비대면 동영상 수업에서는 파이썬 이론 및 예제를 학습하고, 대면 수업 시간에는 코딩 작업을 하며 응용문제를 해결하는 방식으로 진행하였다. 연구 대상에 대한 정보는 표 2와 같이 S1과 S2는 '소프트웨어기초' 과목, C1과 C2는 '컴퓨팅사고' 과목의 분반으로 표시하고 각 분반에 대한 전공영역과 학생 수를 나타내었다.

두 교과목에서는 총 15주의 수업 기간 중 8주 차에는 중간 과제를 작성하고 15주 차에는 기말 프로젝트를 수행하였으며, 각 유형의 작성 방법은 표 3과 같다.

중간 과제에서 제공되는 Type 1과 Type 2는 각 1개씩의 예제 템플릿이 제공되며, 작성 조건은 제공된 템플릿의 기존 콘텐츠 내용을 변경하고 기능을 추가하여 코드를 재구성해야 한다. 단, Type 2에서는 기능 추가를 3개 이상하고 조건문의 중첩구조가 포함되어 있어야 한다. 이때, 동일한 기능을 여러 번 추가한 경우에는 한 개의 기능을 추가한 것으로 간주한다. 기말 프로젝트의 유형인 Type 3은 키오스크 주문하기, 계산기, 온도 및 음량 조절 시스템, 터틀 그래픽으로 그림 그리기 등 총 12개의 예제 템플릿을 제공하고, 콘텐츠를 변경하거나 추가하여 재구성할 수 있다. 기능 추가는 개수에 제한 없이 자유롭게 작성하고 조건, 반복, 함수를 이용한 중첩구조가 포함된다. 예제 템플릿이 제공되지 않는 Type 4는 콘텐츠 계획에서부터 다양한 기능이 포함된 코드 작성까지 자유롭게 작성한다.

표 2. 연구 대상
Table 2. Research subject

Class	Subject	Semester	Major	Number of students
S1	Software Fundamentals	1	Social Sciences	98
S2	Software Fundamentals	2	Social Sciences	80
C1	Computational thinking	1	Humanity	98
C2	Computational thinking	2	Arts and Physical Education	79
Total				355

표 3. 유형별 작성 방법
Table 3. Type-specific constraints

Type	Midterm Assignment		Final Project	
	Type 1	Type 2	Type 3	Type 4
Number of example templates	1	1	12	0
Constraints	Change content	Change content	Change and plan content	Plan content
	Add 2 or more functions	Add 3 or more functions	No restrictions	No restrictions
	Exclude nested-conditional statements	Include nested-conditional statements	Exclude nested-conditional statements	Include nested-conditional statements

Ⅳ. 연구 결과

4-1 예제 템플릿 유형의 선택 변화에 대한 분석

학습자가 선택할 수 있는 중간 과제의 예제 템플릿 Type 1과 Type 2 그리고 기말 프로젝트의 예제 템플릿인 Type 3과 Type 4는 유형별로 사용하는 파이썬 문법에 대한 다양성과 콘텐츠에 따른 문제해결 범위에 서로 차이가 있다.

학기 중반까지 학습하는 파이썬 문법의 입, 출력, 변수, 문자열, 수식, 조건문은 중간 과제 작성에 포함되는 기본 항목인데, Type 1에서 포함되지 않는 조건문의 중첩 구조가 Type

2에는 포함되므로, 콘텐츠에서 문제해결에 적용되는 요구사항들의 범위가 Type 1보가 Type 2가 넓다. 또한, 유형별 작성 방법에서 제시한 작성 조건도 Type 2는 제시된 예제 템플릿의 구조에 3개 이상의 기능을 추가해야 한다. 따라서 Type 2는 Type 1에 비해 난이도가 높다. 또한 Type 3은 파이썬 문법 전체가 포함된 예제이고, Type 4의 작성 방법은 자유롭지만, 설계에서부터 코드 작성까지 모든 것을 창작해야 하므로, 중간 과제에 비해 Type 3와 Type 4의 난이도가 높다. 학습자는 본인 스스로 작성할 수 있는 유형을 선택할 수 있으며, 각 교과목의 분반별 중간 과제 및 기말 프로젝트에서 학습자들의 유형 선택 결과는 표 4와 같다.

표 4. 중간 과제 및 기말 프로젝트 유형 선택 결과
Table 4. Results of selecting midterm and final project types

Class	Midterm Assignment		Number of students	Final Project		Number of students
	Type 1	Type 2		Type 3	Type 4	
S1	41	57	98	18	80	98
S2	37	43	80	17	63	80
C1	44	54	98	22	76	98
C2	33	46	79	16	63	79
Total	155	200	355	73	282	355

이러한 유형 선택 결과에서 중간 과제(Type 1, Type 2)와 기말 프로젝트(Type 3, Type 4)에서의 학생 수 변화를 알아보기 위해, 전체 학생 수(355명)에 대한 표 4에서 계산한 유형별 학생 총수의 비율을 계산하여, 표 5에 %로 표시하였다.

표 5. 유형 선택 비율
Table 5. Percentage of type selection

Type	Midterm Assignment		Number of students	Final Project		Number of students
	Type 1	Type 2		Type 3	Type 4	
Total	155	200	355	73	282	355
%	43.66	56.34	100	20.56	79.44	100

그림 4는 그 결과에 대한 비교 내용을 보여주고 있으며, 유형 선택 비율의 결과를 살펴보면 다음과 같다.

전체 연구 대상자 중 중간 과제에서 Type 1이 43.66%, Type 2는 56.34%로 난이도가 높은 Type 2를 선택한 비율이 12.68% 높았다. Type 1을 선택한 경우, 기능 추가 부분에서 입력과 출력문을 추가한 경우가 많았고, 조건을 확장시켜 소스 코드를 재구성한 경우도 있었다. Type 2에서는 다수개의 입, 출력문과 중첩구조를 포함하기 때문에 조건 기능을 추가한 경우가 많았다. 기말 프로젝트에서는 Type 3가 20.56%, Type 4는 79.44%로 Type 4의 선택 비율이 58.88% 높았으며, 난이도가 높은 유형을 선택한 비율이 중간 과제에 비해 23.10% 증가되었다. Type 3에서는 주제별로 총 12개의 예제 템플릿이 제공되었기 때문에, 학습자들은 각 예제 중 한 개 이상을 참조하여 콘텐츠를 기획하고 코드를 재구성하여 프로젝트를 완성하는 것이 관찰되었다. Type 4를 선택한 경우, 프로젝트의 주제를 선정 후 콘텐츠 기획에서부터 코드의 구조화 및 코딩 작업까지 새롭게 작성하였다.

이와 같은 결과는 중간 과제에서 학습자는 프로그래밍이 동작하는 기본적인 흐름을 예제 템플릿을 통해 익히며, 학습자 스스로 필요한 기능을 생각하고 코드로 추가하며 재구성하는 작업을 경험하고, 기말 프로젝트에서는 이를 바탕으로 학습자 스스로 코드의 구조를 설계하여 콘텐츠 기획부터 파이썬의 다양한 기능을 이용하여 프로그램을 완성할 수 있다는 것을 보여준다.

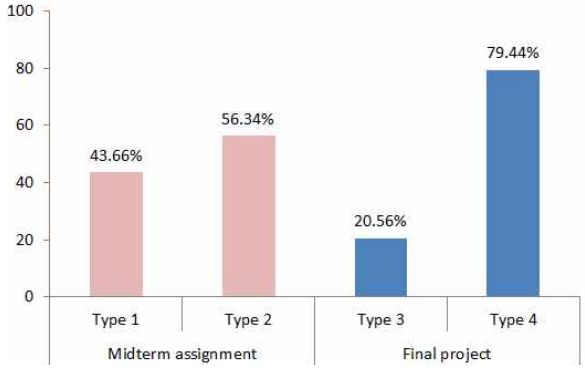

그림 4. 유형 선택 결과의 비교
Fig. 4. Comparison of type selection results

4-2 기말 프로젝트 결과의 기능성 분석

기말 프로젝트는 학습자 스스로 문제를 정의하고, 프로그램의 흐름을 이해하며, 결과를 도출하여 문제를 해결하는 방법으로 진행하게 된다. 이 과정에서 학습자는 콘텐츠를 기획하고, 프로그램을 만들기 위해 필요한 파이썬의 기능을 판단하고 선택하여 사용하게 된다. 이처럼 다양한 주제에 대해 완성도가 있는 프로젝트를 수행하기 위해, 파이썬의 기본 기능부터 확장 기능까지 프로그램의 각 용도에 맞게 필요한 기능을 선택하여 전체적인 구조에 적절하게 적용하는 것은 중요한 부분이다.

표 6은 기말 프로젝트 수행에서 학습자가 사용한 파이썬의 기능별 사용 빈도와 비율을 나타낸다. 이때 각 기능을 여러 번 사용한 경우에는 한 번 사용한 것으로 간주했다. 터틀 그래픽만을 사용한 경우에는 이미지 모양을 그리는 동작의 시작 부분을 입력, 이미지가 그려진 부분을 출력으로 인정했다.

파이썬 기능별 사용 내용을 살펴보면, 입력과 출력은 프로그램의 기본적인 요소이기 때문에 모든 학습자가 프로젝트 작업에서 사용하였다. 입력문이나 수식에 많이 사용되는 변수는 95.21%였고, 터틀 그래픽만을 이용한 프로젝트에서는 변

수를 사용하지 않은 경우가 있었다. 94.93%의 학습자가 사용한 문자열의 경우에도 입, 출력문에서 많이 사용되지만, 변수와 마찬가지로 터틀 그래픽 작업에서는 제외된 경우가 있었다. 이 외에도 수식(95.49%)과 조건문(90.99%)도 콘텐츠와 연관성이 없거나 터틀 그래픽에서 사용하지 않은 경우가 있었다. 중첩구조는 75.21%로 중간 과제에 비해 증가되는 현상을 보였는데, 이는 조건의 중첩이나 조건과 반복의 중첩을 통해 복잡한 문제를 해결한 경우였다. 콘텐츠에 따라 여러 개의 데이터를 다룰 수 있는 리스트의 경우 45.92%의 사용 비율을 보였다. 또한 난이도가 높은 확장 기능인 반복문(85.63%), 함수(85.07%), 모듈(90.42%)의 경우에도 비교적 높은 사용 비율을 보였다.

이러한 결과로부터 학습자가 각 기능의 역할을 정확히 알고, 필요한 부분에 맞게 적용하여 원하는 결과물을 만들어 낸 것을 알 수 있었다. 특히 함수와 모듈과 같이 확장 기능을 사용하여 코드를 간결하고 효율적으로 작성한 점도 관찰되었다.

표 6. 파이썬 기능별 사용 빈도 및 비율
Table 6. Frequency and percentage of usage for each function

Function	S1	S2	C1	C2	Total	%
Input/Output	98	80	98	79	355	100
Variables	98	80	83	77	338	95.21
Strings	93	75	94	75	337	94.93
Python formula	93	74	95	77	339	95.49
Conditional Statements	92	76	87	68	323	90.99
Nested-conditional statements	73	63	69	62	267	75.21
Lists	44	39	44	36	163	45.92
Loops	82	68	85	69	304	85.63
Functions	85	72	78	67	302	85.07
Modules	91	66	94	70	321	90.42

4-3 학습자 의견 분석

수업에 참여한 학습자들의 중간 과제와 기말 프로젝트 수행 과정에서 어려웠던 점에 대한 의견을 살펴보기 위해 설문조사를 하였다. 설문 조사는 중간 과제와 기말 프로젝트 작성 후에 진행하였고, 프로그램을 작성하는 기본 과정인 주제 설정, 알고리즘 설계, 코딩, 오류수정 중 어려웠던 작업을 중복으로 선택할 수 있게 하였다. 표 7과 표 8은 중간 과제와 기말 프로젝트에 대한 조사 결과를 나타낸다. 표에서 N은 각 작업을 선택한 학습자 수이고, ()안의 %는 다중 응답 결과를 비교하기 위한 케이스 퍼센트(%)로 분반별 전체 응답자 중 각 작업을 선택한 학습자의 비율을 나타낸다. 마지막 행의 전체의 경우, 학습자 수의 합과 케이스 퍼센트의 평균을 정리하였다.

표 7. 중간 과제에서 어려웠던 작업
Table 7. Difficult task in midterm assignment

Class	Subject	Algorithm	Coding	Error Correction
	N(%)	N(%)	N(%)	N(%)
S1	35 (35.71)	78 (79.59)	85 (86.73)	16 (16.33)
S2	34 (42.50)	69 (86.25)	71 (88.75)	13 (16.25)
C1	47 (47.96)	88 (89.80)	92 (93.88)	19 (19.39)
C2	33 (41.77)	61 (77.22)	65 (82.28)	7 (8.86)
Total	149 (41.99)	296 (83.22)	313 (87.91)	55 (15.21)

표 8. 기말 프로젝트에서 어려웠던 작업
Table 8. Difficult task in final project

Class	Subject	Algorithm	Coding	Error Correction
	N(%)	N(%)	N(%)	N(%)
S1	66 (67.35)	58 (59.18)	62 (63.27)	37 (37.76)
S2	53 (66.25)	49 (61.25)	50 (62.50)	27 (33.75)
C1	73 (74.49)	65 (66.33)	63 (64.29)	35 (35.71)
C2	43 (54.43)	43 (54.43)	45 (56.96)	29 (36.71)
Total	235 (65.63)	215 (60.30)	220 (61.76)	128 (35.98)

중간 과제 작성 과정에서 가장 어려웠던 작업은 총 355명의 학습자 중 313명(88.91%)이 선택한 코딩 작업과 296명(83.22%)이 선택한 알고리즘 작업이 높은 분포를 보였다. 그 다음으로는 149명(41.99%)이 선택한 주제 선정이었고, 오류수정은 55명(15.21%)이 선택하였다. 기말 프로젝트에서는 주제 선정이 235명(65.63%)으로 가장 높은 분포를 보였고, 코딩 작업 220명(61.76%), 알고리즘 작업 215명(60.30%), 오류수정 128명(35.98%) 순이었다.

중간 과제와 기말 프로젝트 수행 중 어려웠던 점에 대한 변화를 살펴보면, 중간 과제를 작성하며 가장 어렵게 느꼈던 코딩 작업은 기말 프로젝트에서는 26.15%가 감소하였고, 알고리즘 작업의 경우에도 22.92%가 감소되는 것이 관찰되었다. 이러한 변화는 컴퓨터 비전공자가 프로그래밍 과제를 작성해 본 경험이 부족하기 때문에 중간 과제에서는 코딩 작업과 알고리즘을 설계하는 방법이 어려웠지만, 예제 템플릿을 이용한 중간 과제 작성 과정의 전반적인 경험이 기말 프로젝트 수행에 있어 도움이 되었을 것으로 해석된다. 반면 주제 선정 작업은 어려웠던 점에 대한 의견이 중간 과제 때보다 23.64% 증가되었다. 이는 단일 예제 템플릿을 사용한 중간 과제와는 달리, 기말 프로젝트에서는 다수 개의 예제 템플릿을 참고하거나 학습자 스스로 새롭게 응용 프로그램을 설계해야 하므

로 주제 선택의 폭이 넓고 다양해지면서 주제에 맞는 프로그램 구성의 어려움 때문인 것으로 보인다. 오류수정의 경우에는 중간 과제에 비해 사용한 문법의 종류가 증가하고 높은 난이도의 확장 기능을 사용하기 때문에, 발생하는 오류가 증가하며 수정 작업에도 많은 시간이 소요되는 어려움이 있다. 이로 인해 어려웠던 점에서 오류에 대한 의견이 중간 과제 때보다 기말 프로젝트에서 20.77% 증가된 것으로 파악된다.

표 9은 본 연구에서 제안한 방법으로 학습한 학습자들의 코딩에 대한 이해력과 실력 그리고 응용력 및 창의력 향상에 대한 의견을 살펴보기 위해 학기 말에 조사한 설문 내용이다. Q1과 Q2는 중간 과제, Q3과 Q4는 기말 프로젝트를 수행과 관련된 내용이며, 그림 5에는 각 설문 내용을 5점 척도로 구분하여 조사한 결과의 평균값으로 나타내었다.

중간 과제를 수행 후 코딩 방법에 대한 이해력이 향상되었다는 Q1에 대해 각 분반(S1, S2, C1, C2)에 대한 전체 평균이 4.19점으로 긍정적인 의견을 보였다. 또한 중간 과제 작성 경험이 기말 프로젝트 수행에 도움이 되었다는 Q2에 대한 의견도 전체 평균이 4.71점으로 높은 점수를 나타냈다. 이러한 결과는 학습자들이 중간 과제에서 경험한 과제 작성 방법이 학습 범위를 확장하여 진행하는 기말 프로젝트 수행에 도움이 된 것으로 볼 수 있다. 기말 프로젝트와 관련된 설문 중 코딩 실력 향상에 대한 질문 Q3는 전체 평균 4.47점으로 코딩에 대한 이해력 향상과 관련된 질문 Q1의 결과에 비해 0.28점 상승한 값이 나왔다. 즉, 제안한 학습 방법이 학습자들의 코딩에 대한 이해력과 함께 학기 말에는 코딩 실력 향상에 도움이 된 것으로 볼 수 있다. 또한 응용력과 창의력 향상에 대한 질문인 Q4에서도 전체 평균 4.63점의 높은 점수가 나타난 것은 기말 프로젝트 작업이 학습자의 응용력과 창의력을 이끌어 내는 데 있어 긍정적인 작용을 한 것으로 볼 수 있다.

학습자들이 기말 프로젝트를 진행하며 최종 보고서의 수행 소감에 작성한 의견을 도움이 되었던 점과 어려웠던 점으로 요약해 보면 다음과 같다. 도움이 되었던 점은 “현실에서 간단하게 사용하는 자동화 시스템을 직접 계획해 보니 고려해야 할 문제들이 많은 것을 알게 되었고, 응용 프로그램을 직접 만들었다는 사실이 신기하고 뿌듯하다.”, “계획한 주제를 구현하기 위해 파이썬의 다양한 기능을 적용하여 결과물을 완성할 수 있어서 만족스러웠다.”, “파이썬의 여러 기능을 사용하기 위해 수업 시간에 배운 내용을 복습할 수 있었고, 어려웠던 부분도 직접 작성해 보니 이해가 되었다.”와 같은 의견이 있었다. 이와 같이 기말 프로젝트 작성 과정에서 학습자들이 한 학기 동안 학습한 내용을 잘 이해하고 스스로 결과물을 완성해 내는 적극적인 의지를 확인 할 수 있다.

반면, 어려웠던 점은 “중간 과제에 비해 기말 프로젝트에서는 주제에서부터 파이썬을 다양하게 적용하는 기능적인 부분까지 스스로 결정해야 할 문제들이 많아서 시간이 오래 걸리고 어려웠다.”, “코딩을 하며 발생하는 오류를 해결하는 작업이 어렵고 시간이 오래 걸렸다”, “수업 시간에 배운 내용만으로 구현하지 못한 내용이 있어서 아쉽지만, 향후에 파이썬을 더 공부해서 만들어 보고 싶다.”와 같은 의견이 있었다. 이와 같이 기말 프로젝트 수행 과정에서 오랜 시간 동안 학습자들이 고민하고 반복 작업하는 과정을 경험하며 결과물 완성을 위해 노력한 것을 알 수 있다.

표 9. 설문 내용
Table 9. Survey contents

Question Number	Question Contents
Q1	After completing the midterm assignment, my understanding of coding methods improved.
Q2	The experience of writing midterm assignments helped me carry out my final project.
Q3	My coding skills improved while working on my final project.
Q4	After completing the final project, my application skills and creativity improved.

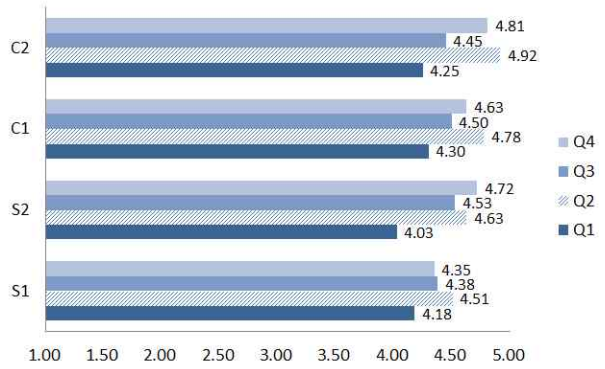

그림 5. 설문 조사 결과
Fig. 5. Survey results

4-4 종합 분석 및 기대효과

앞서 언급한 학습자의 중간 과제 및 기말 프로젝트 작성에 대한 분석 내용과 학습자 의견을 종합적으로 정리해 보면 다음과 같다.

본 연구에서 제안한 방법을 통해 첫째, 학습자는 난이도가 다른 유형별 예제 템플릿과 코드 수정 및 재구성 원리를 적용한 학습을 통해 점진적으로 코딩 작성 능력이 향상되는 긍정적인 결과를 보였다. 둘째, 기말 프로젝트 작업에서는 프로젝트의 시작부터 완성까지 학습자 스스로 진행해 나가며 필요한 기능에 적합한 파이썬의 다양한 문법을 선택하여 효율적이며 적절하게 적용할 수 있는 능력이 향상되는 것을 확인할 수 있었다.

수업에 참여했던 학습자의 의견은 첫째, 중간 과제에서 다양한 문제를 고려한 주제를 계획하는 알고리즘 작업과 파이썬 언어를 이용한 코딩 작업이 어려웠다는 의견이 많았지만, 중간 과제 작성 경험을 바탕으로 기말 프로젝트에서는 두 작업에 대해 느끼는 어려움이 낮아지는 의견을 보여, 학습자들

이 문제를 계획하고 코딩하는 작업에 익숙해진 것이 관찰되었다. 반면, 중간 과제에 비해 학습자의 자율성이 폭넓어진 기말 프로젝트에서는 창의적인 생각을 통해 주제를 선정하는 부분과 복잡하고 심화된 파이썬 기능을 사용하는 과정에서 발생하는 오류 해결이 어려웠다는 의견이 증가된 결과를 보였다. 이러한 설문 조사 결과로부터 컴퓨터 비전공자가 점진적으로 코딩에 익숙해지며 복잡한 구조의 기능을 포함한 프로젝트 결과물을 스스로 완성해 나갈 수 있다는 것을 알 수 있다. 둘째, 중간 과제에서 예제 템플릿과 코드의 재구성 작업을 하며 코딩의 원리를 이해하고, 기말 프로젝트에서는 스스로 콘텐츠를 계획하고, 프로그램 구조를 설계하여 응용 프로그램을 완성해 나가며, 코딩 실력과 응용력 및 창의력이 향상되었다는 의견을 보였다. 이는 제안한 방법이 정해진 문제를 특정 기능을 사용하여 풀어내는 시험방식과는 달리 학습자가 스스로 할 수 있는 내용을 선택하고 학습한 내용을 이해하며 프로그램의 기능을 점진적으로 확장해 나갈 수 있도록 구성되어 있기 때문에, 기말 프로젝트를 수행할 때보다 쉽고 효율적으로 접근할 수 있었다고 해석된다. 셋째, 학습자들의 기말 프로젝트 수행 소감은 중간 과제에 비해 학습자 스스로 사고하고 다양한 방법을 고려한 문제를 계획하여 결과물을 완성하는 과정에서 성취감을 느낀 것으로 파악된다. 반면, 한 학기 동안 학습한 폭 넓은 범위의 파이썬 문법을 적용하여 스스로 계획한 프로젝트 작업을 하기 위해 긴 시간 동안 어렵고 복잡한 코딩 작업을 하며 결과물을 완성한 것으로 관찰된다.

본 연구에서 제안한 학습모델은 컴퓨터를 전공하지 않거나 코딩에 익숙하지 않은 학습자들에게 적용할 경우 다음과 같은 효과를 기대할 수 있다. 첫째, 코딩 학습 측면에서 학습자가 프로그래밍 문법을 단순히 암기하고 주어진 문제에 대해 코딩하는 학습 방법과는 달리, 중간 과제에서 제공된 예제 템플릿으로 프로그램의 구조를 익히고 코드의 재구성 과정에서 프로그래밍 문법을 자연스럽게 이해할 수 있다. 둘째, 프로젝트 제작 관점에서 텍스트 기반 프로그램의 동작 원리에 따라 코드의 기능을 변경하고 보완하는 과정을 통해 학습자 스스로 계획한 문제를 해결해 나갈 수 있다. 셋째, 초보 학습자의 점진적인 코딩 학습 강화와 확장을 위해, 교수자는 난이도별 예제 템플릿에 대한 제약 조건을 설계하고, 이 예제 템플릿들을 이용하여 학습자가 학습한 문법을 종합적으로 적용할 수 있는 기반을 마련해 줄 수 있다. 넷째, 소프트웨어 프로젝트에 대한 경험이 없는 학습자에게 프로젝트를 수행하는 데 필요한 전반적인 과정을 경험하게 하여 자동화 시스템에 대한 이해력을 키워줄 수 있다.

V. 결 론

본 연구는 프로그램 초보 학습자들이 쉽게 프로젝트를 작성할 수 있도록 예제 템플릿과 코드의 재구성 학습 방법을 기반으로 프로그래밍 학습모델을 설계하고 실제 수업에 적용하였다. 제안한 방법은 프로그래밍 초보자가 프로젝트를 수행하는 과정에서 느끼는 어려움을 고려하여 프로그램의 개별적인 문법을 익힌 후, 이들을 조합하여 표현하는 방법, 입력과 출력 관계를 고려한 프로그램의 구조 설계, 프로그램의 필요 기능을 확장하는 방법을 학습 과정에 반영하였다. 학습자들에게 이러한 학습 방법을 통해 확장성과 다양성을 고려한 코딩 프로젝트를 경험하게 할 수 있고, 코딩에 대한 자신감과 향후 코딩과 관련된 연장 학습에 동기를 부여해 줄 수 있다. 중간 과제에서는 예제 템플릿을 기반으로 프로그램의 입력과 출력의 흐름과 개별 문법을 조합하여 설계된 프로그램의 구조를 익히고, 필요한 기능 또는 본인의 아이디어로 도출한 기능을 추가하여 프로그램을 재구성하는 방법을 연습한다. 그리고 기말 프로젝트에서는 전반적인 프로젝트 수행 과정(문제의 계획, 문제해결을 위한 알고리즘 설계, 프로그래밍 구현 및 테스트)을 경험하게 하며, 난이도가 높지만 활용성이 다양한 파이썬 프로그램의 확장 기능을 학습자 스스로 선택하여 응용해 볼 수 있다. 이러한 프로젝트 수행 방법을 통해 학습자가 일상생활에서 사용하는 자동화 시스템을 이해하고, 본인의 아이디어를 적용한 응용 프로그램을 직접 만들어 볼 수 있는 경험을 할 수 있다. 그리고 학습자의 예제 템플릿 유형 선택 결과 분석, 기말 프로젝트에서의 기능성 분석, 학습자 의견 분석 등을 통해, 제안한 학습모델이 컴퓨터 비전공자에게 효율적으로 적용할 수 있음을 확인하였다.

이와 같은 경험은 컴퓨터 비전공자인 학습자에게 향후 연계된 학습에 대한 동기부여를 할 수 있고, 자동화 시스템에 대한 이해도와 친밀도를 높일 수 있으며, 이를 활용하거나 개발하는 데 있어 무한한 능력이 있다는 자신감과 함께 지속적으로 변화되는 디지털 세상에 적응력을 키우는 데 도움이 될 수 있을 것이다. 이러한 연구 결과를 바탕으로 향후에는 컴퓨터 비전공자들에게 보다 실용적이고 현실적인 코딩 학습을 경험하게 하기 위한 연구를 진행할 계획이다.

감사의 글

본 연구는 2024학년도 경기대학교 학술연구비(일반연구과제) 지원에 의하여 수행되었음

참고문헌

[1] M. Lee, “Exploring the Effect of SW Programming Curriculum and Content Development Model for Non-Majors College Students: Focusing on Visual Representation of SW Solutions,” *Journal of Digital Contents Society*, Vol. 18, No. 7, pp. 1313-1321, November

2017. https://doi.org/10.9728/dcs.2017.18.7.1313

[2] H.-W. Jung, “A Case Study of Python Programming Error in an Online Learning Environment,” *The Journal of the Convergence on Culture Technology*, Vol. 7, No. 3, pp. 247-253, August 2021. https://doi.org/10.17703/JCCT.2021.7.3.247

[3] H.-W. Jung, “A Study on Teacher-Learner Feedback Method for Effective Software Project Execution of Non-Computer Major Students,” *The Journal of the Convergence on Culture Technology*, Vol. 5, No. 1, pp. 211-217, February 2019. https://dx.doi.org/10.17703/JCCT.2019.5.1.211

[4] M. Romero, A. Lepage, and B. Lille, “Computational Thinking Development through Creative Programming in Higher Education,” *International Journal of Educational Technology in Higher Education*, Vol. 14, 42, December 2017. https://doi.org/10.1186/s41239-017-0080-z

[5] P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, “Motivating Project-Based Learning: Sustaining the Doing, Supporting the Learning,” *Educational Psychologist*, Vol. 26, No. 3-4, pp. 369-398, 1991. https://doi.org/10.1080/00461520.1991.9653139

[6] P. Blumenfeld, B. J. Fishman, J. Krajcik, R. W. Marx, and E. Soloway, “Creating Usable Innovations in Systemic Reform: Scaling up Technology-Embedded Project-Based Science in Urban Schools,” *Educational Psychologist*, Vol. 35, No. 3, pp. 149-164, 2000. https://doi.org/10.1207/S15326985EP3503_2

[7] T. Markham, J. Larmer, and J. L. Ravitz, *Project Based Learning Handbook*: *A Guide to Standards-Focused Project Based Learning for Middile and High School Teachers*, 2nd ed. Novato, CA: Buck Institute for Education, 2003.

[8] 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, February 2018. https://doi.org/10.15207/JKCS.2018.9.2.007

[9] C. M. Nam and C. W. Kim, “Development of Computational Thinking Based Coding_Projects Using the ARCS Model,” *Journal of The Korean Association of Information Education*, Vol. 23, No. 4, pp. 355-362, August 2019. https://doi.org/10.14352/jkaie.2019.23.4.355

[10] D. Koo, “Development of Digital Storytelling Education Program Based on Software Programming,” *The Journal of Korea Elementary Education*, Vol. 25, No. 1, pp. 245-260, March 2014. https://doi.org/10.20972/kjee.25.1.201403.245

[11] S.-W. Ha and K.-H. Huh, “A Curriculum for Mobile Programming Education that Includes a Project Completion and It’s Implementation Results,” *Journal of the Korea Society of Computer and Information*, Vol. 21, No. 9, pp. 139-147, September 2016. http://dx.doi.org/10.9708/jksci.2016.21.9.139

[12] M. Tsur and N. Rusk, “Scratch Microworlds: Designing Project-Based Introductions to Coding,” in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE ’18)*, Baltimore: MD, pp. 894-899, February 2018. https://doi.org/10.1145/3159450.3159559

[13] M. Frydenberg and K. Mentzer, “From Engagement to Empowerment: Project-Based Learning in Python Coding Courses,” *Information Systems Education Journal*, Vol. 19, No. 3, pp. 47-62, June 2021.

[14] A. J. Lekan and O. S. Abiodun, “Children Perceptions of the Effectiveness of Online Coding as a Supplement to in-person Boot Camps,” *International Journal of Scientific Advances*, Vol. 1, No. 3, pp. 187-191, November-December 2020. https://doi.org/10.51542/ijscia.v1i3.12

[15] T. Kim and S. Han, “Development of Python Education Program for Block Coding Learners,” *Journal of the Korean Association of Information Education*, Vol. 22, No. 1, pp. 53-60, February 2018. http://dx.doi.org/10.14352/jkaie.2018.22.1.53

[16] Python Software Foundation. The Python Tutorial [Internet]. Available: https://docs.python.org/3.12/tutorial/index.html.

[17] W3Schools. Python Tutorial [Internet]. Available: https://www.w3schools.com/python.

정혜욱(Hye-Wuk Jung)

2005년 : 성균관대학교 정보보호학과 (공학석사)
2013년 : 성균관대학교 컴퓨터공학과 (공학박사)
2015년~2018년: 성균관대학교 컨버전스연구소 선임연구원
2018년~현 재: 경기대학교 자유교양대학 교양학부 부교수
※관심분야 : 코딩 교육, 프로그래밍 언어, 인공지능 등