http://www.doi.org/10.32823/jcic.7.2.202105.69

창의정보문화연구 제7권 제2호 (2021. 5) ISSN 2384-2008(Print) / 2733-8681(Online)

비전공자 대상의 컴퓨터 프로그래밍 입문 교양 수업에서의 학습자의 어려움 분석

김재경, 손의성
연세대학교

요약

과거 프로그래밍은 컴퓨팅 관련 전공자들을 대상으로 일부의 학생들이 배우는 과목이었으나, 오늘날 소프트웨어 중심 산업이 도래함에 따라 컴퓨팅사고 교육과 함께 컴퓨팅 문제 해결의 도구로서 모든 전공 분야의 학생들에게 교육이 이루어지고 있다. 그러나 컴퓨팅 입문 과목을 필수로 이수해야 하는 학습자들은 익숙하지 않은 컴퓨팅 문제 해결 방식과 프로그래밍 언어라는 새로운 내용에 학습에 큰 어려움을 겪으며 학습 효과, 자신감, 흥미 저하와 같은 부정적인 현상으로 이어질 수 있다. 본 논문에서는 비전공자들이 프로그래밍 언어 과목을 학습하면서 겪는 어려움의 원인을 설문 조사, 일지 및 성취도를 질적 및 양적 연구로 분석하여 파악하고, 이를 최소화할 수 있도록 방안을 제시하여 향후 어려움을 최소화하는 교육 방안의 설계에 도움이 되고자 한다.

■ 중심어 : 컴퓨터 프로그래밍, 컴퓨팅 문제 해결, 어려움, 비전공자, 컴퓨팅 사고

Difficulty Analysis of an Introductory Computer Programming Course for non-Major Students

Jaekyung Kim, Eisung Sohn
Yonsei University

Abstract

In the past, computer programming was a course taken by students of computing domain majors. With the advent of the fourth industrial revolution, students in all major fields are taking it as the general required course. However, students have difficulties in learning new subject such as unfamiliar computational problem solving approach and general purposed programming language, which can lead to negative phenomena such as learning effectiveness, confidence, and decreased interest. In this paper, the causes of difficulties experienced by non-majors students while learning programming language are analyzed and identified through qualitative and quantitative research on questionnaires, journals, and achievements. Thus, we suggest that designing an educational plan that minimizes difficulties.

■ Keyword : computer programming, computational problem solving, difficulty, non-major students, computational thinking

접수일자 : 2021. 3. 29. 심사완료일자 : 2021. 5. 6. 게재확정일자 : 2021. 5. 31.
교신저자 : 김재경(E-mail : kim.jk@yonsei.ac.kr)
* This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (2019R1G1A1010839).

Ⅰ. 서 론

산업과 경제의 기반이 소프트웨어 기술로 빠르게 변화한 4차 산업에서 컴퓨팅사고(computational thinking), 인공지능, 딥러닝(deep learning), 블록체인(bloack chain) 등과 같은 단어들은 특정 전문가나 전공 분야에서 사용하는 전문 용어가 아니라 일상생활로 이미 깊게 들어온 일반적인 개념이 되었다. 이와 같은 산업의 변화에 대응하고자 국내외 교육기관에서는 전공 분야를 막론하고 정보통신기술 및 소프트웨어를 모든 이들에게 교육하여 학생들이 미래의 사회의 원동력이 될 수 있도록 노력하고 있다.

먼저 국내의 경우, 2015년도에 발표된 개정 교육과정에서 2018년도부터 정보 교과를 중등 교육과정에서 적용하고, 단계적으로 2019년도에는 초등 교육과정에서도 교육하고 있다[1,2]. 또한 2020년 3월에 교육부와 과학기술정보통신부는 “2020년 소프트웨어 선도학교 및 AI 시범학교”를 실시하고 전국의 2,011개 초·중등학교를 소프트웨어 교육 선도학교를 지정하였으며 이 중 247개 학교는 인공지능 교육 과정을 시범 운영하고 있다. 이와 함께 과학기술정보통신부에서는 보도자료를 통하여 모든 학생이 내실 있는 기초 소양을 기를 수 있도록 소프트웨어 교육을 받을 수 있도록 정책을 추진할 계획이라고 밝혔다[3].

미국의 경우 연방 5개년 STEM 교육 전략 계획(Federal 5-Year STEM Education Strategic Plan)을 2018년도에 수립하고 향후 5년 동안 매년 최소 약 2,500억원을 투자하기로 하였으며, 나아가 2019년도에는 미국 AI 계획(The American AI initiative)를 추진하여 전문인력 양성 및 전 국민 STEM 교육 확대를 꾀하고 있다[4].

영국의 경우에도 2014년에 컴퓨팅 교육 의무화 이후, 2018년도에는 산업 전략 인공지능 분야 합의(Industry Strategy Artificial Intelligence Sector Deal)를 통하여 약 145억원의 장학금과 STEM 교육에 약 6천억원을 투자하고 있다[5].

이와 같은 정보통신기술의 발전 및 인력 창출을 위해 우리나라 뿐 아니라 세계 각국은 다양한 방식으로 교육 분야에서 치열한 경쟁을 하고 있다. 따라서 정보통신기술이 중심이 되는 산업 사회에 대응할 수 있는 융합 인재 양성을 위한 교육에 대한 고찰이 필요하다[6]. 특히 소프트웨어 교육에서 가장 기초가 되는 프로그래밍 교육의 방향과 효용성에 대해 다양한 각도에서 연구가 필요하다.

그러나 고등 교육과정에서 학습자가 겪는 어려움에 대한 연구는 아직 초기 단계이며, 주로 과목의 설계, 교육 모델의 제안[7], 혹은 컴퓨팅사고력 향상[8-10]에 대한 연구가 주를 이루고 있다. 기존의 컴퓨터과학 전공자를 대상으로 한 교육과 달리, 비전공자는 프로그래밍 언어의 학습 과정에서 많은 어려움을 겪기 때문에 이들이 경험하는 구체적인 문제점들을 다각도로 분석하고 해결방안을 마련해야만 효율적인 컴퓨팅 기초 교육이 이루어질 수 있다.

본 논문에서는 대학에서 컴퓨터 관련 비전공생등을 대상으로 하는 범용 프로그래밍 언어(GPL) 교양과목에서 입문자들이 경험하는 어려움을 구체적으로 분석하고 원인을 파악하도록 하며, 어려움을 최소화 할 수 있는 방안을 제시하여 추후 교육 전략 및 컨텐츠 개발에 필요한 가이드를 마련하도록 한다. 이를 위하여 학습자들이 느끼는 어려움을 설문조사와 일지를 통하여 질적으로 분석하고, 학습 성취도를 수강 동기와 계열별 변인으로 양적 분석하도록 한다.

Ⅱ. 관련 연구

Konecki[11]는 여러 가지 개념을 기호화 하여 표현하는 프로그래밍의 추상적인 특징은 대부분의 학습자들이 어려움을 호소하는 부분이며 관련 수업에서 많은 실패 사례들이 보고된다고 하였다. 이는 인간의 사고와 컴퓨터 알고리즘 표현 방식 간에는 큰 차이가 있기 때문에 입문자의 경우 자신이 생각한 논리를 올바른 프로그래밍 코드로 표현하기가 어렵기 때문이라고 주장하였다. 또한 입문자들은 프로그래밍 언어의 새로운 문법과

기능을 익히는데 많은 시간을 투자하는 경향이 있어 상대적으로 문제 해결을 위한 알고리즘 설계에서 어려움을 겪는다고 하였다.

Bosse[12]는 프로그래밍 교육에서의 어려움을 학습자와 교수자 관점 등으로 구분하여 분석을 하였는데 학습자는 문법 에러, 잘못된 변수 사용, 의미적 에러의 해결 등에서 어려움을 겪었고, 반면에 교수자는 논리적 추론(logical reasoning) 개념을 학습시키는데 어려움을 호소하였다. 교수자가 문제해결력을 훈련시키기 위해 의사코드(pseudocode)나 흐름도(flowchart)를 제공하고 이를 프로그래밍 명령어로 변환하는 방식을 기대하였으나, 학습자는 이를 무시하고 문제 풀이를 곧바로 프로그래밍 코드 작성부터 시작하는 경향이 있었다.

이철현[13]은 예비 초등예비교사들이 교육용 프로그래밍 언어인 스크래치를 학습할 때 겪는 어려움을 분석하였다. 블록 프로그래밍 언어에서 코드 작성은 상대적으로 쉽게 하였으나 문제 분석과 모듈화 및 디버깅을 가장 어려워하였다. 또한 프로그래밍에 대한 선호도가 낮은 경우 어려움을 크게 느끼는 것으로 나타났다.

김수환[14]은 비전공 학습자들이 컴퓨팅사고 교육에서 겪는 어려움을 조사하였으며 스크래치 도구를 활용한 수업에서 재미, 흥미 및 수업 난이도에 대한 인식의 변화와 컴퓨팅사고 교육에 대한 인식이 어떻게 변화하였는가를 분석하였다.

성정숙[15]은 스크래치에서 입문자가 겪는 상황과 행동에 대해 분석을 하였다. 이 연구에서는 인문계열 학생들의 프로그래밍에 대한 흥미도와 자신감과 같은 심리적인 요인을 설문과 인터뷰 등을 통해 분석하였으며, 이를 통하여 비전공 학생들이 학습하는 프로그래밍 과목은 컴퓨터 전공 과정의 학생들을 위한 교육과는 다른 방법으로 이루어져야 하며 학습자에 대한 면밀한 관찰이 필요하다고 하였다. 이 외에도 최정원은 프로그래밍 교육용 사이트인 code.org에 포함된 순차, 반복, 논리 및 이벤트의 개념을 분류하여 학습자의 어려움을 조사하였다.

선행 연구에서는 비전공자를 대상으로 한 대학 수업에서 스크래치 프로그래밍이 학습자의 동기나 흥미와 같은 심리적 요소와 문제해결력에 미치는 영향을 분석하였으며 스크래치 언어의 특성상 문법적인 요소보다는 변수, 제어문, 함수와 같이 개념적인 범주를 기준으로 문제점을 파악하였다.

본 연구에서는 범용 프로그래밍 언어(GPL) 언어 교육에서 발생할 수 있는 세부적인 어려움 요소를 정의하고 수업 진도에 따른 컴퓨팅 문제 해결의 어려움을 분석하였다. 이를 통하여 학습자들이 실제로 겪는 프로그래밍의 어려움과 실수를 최소화할 수 있는 지침을 제시한다.

Ⅲ. 연구 방법

3.1. 연구 대상

연구 대상은 2020학년도 1학기에 개설된 1학년 대상 Python 프로그래밍 입문 교양 강좌의 수강생으로 1학년 학생 159명이다. 자연・공학 계열의 학생은 59명, 인문・사회 계열 및 그 외 계열의 학생은 100명으로 집계되었으며 수강 대상은 컴퓨터 관련 전공 학생과 동일 학기에 다른 프로그래밍 관련 과목 수강생을 제외하였다. 수업은 1학기 동안 주차별 2시간의 이론 수업 후 2시간의 실습수업으로 진행되었으며 표 1과 같은 수업 내용과 방법으로 진행되었다.

<표 1> 수업 주제와 실습 진행 방식
<Table 1> Course Subject and Processing Method

순번	주제	진행방식
1	Input/Output	실습/설문조사/일지
2	Data Types	실습/일지
3	Conditional Structure	실습/일지
4	Iterative Structure	실습/일지
5	List I	실습/일지
6	List II	실습/일지
7	Function	실습/일지
8	Dictionary	실습/일지
9	String & Text File	실습/일지
10	Module & Class	실습/일지

3.2. 연구 방법

본 논문에서는 학습자들의 의견과 실제 성취도를 분석하며 보다 실증적인 어려움을 파악하기 위해 Creswell[17]의 6가지 혼합적 분석 모델 중 동시적 삼각화 전략(concurrent triangulation strategy)으로 질적 자료와 양적 자료를 동시에 수집하여 분석하였다. 삼각화 모델은 가장 많이 사용되는 전략으로 질적 자료와 양적 자료를 동시에 수집하여 두 방법의 분석 결과를 얻을 수 있는 장점이 있다[18].

먼저 학습자들이 겪는 어려움을 질적 분석으로 파악하기 위해서 설문조사 및 관찰 일지를 수집하고 내용 분석을 통하여 요소와 범주를 정의하였다. 설문조사는 학기 초기에 '본 수업의 수강 동기는 무엇인가요'를 묻는 주관식 문항으로 1회 시행하였으며, 관찰 일지는 10번의 실습 시간에 문제 해결 시 경험하는 어려움에 대한 학습자들의 의견을 기록하여 수집하고 내용 분석을 통하여 범주화하고 세부 요소를 설정하였다. 그 결과 표 2와 같은 결과를 얻을 수 있었으며 작성된 범주와 요소에 대해 분석을 수행하였다.

<표 2> 내용 분석 결과
<Table 2> Result of Content Analysis

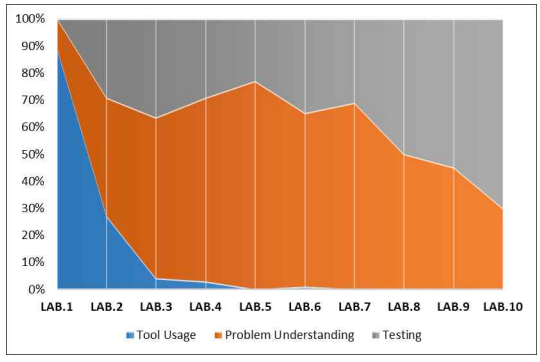
범주	요소	내용
학습의 어려움	컴퓨팅 문제해결 방식	새로운 도구 사용
		문제의 이해
		결과 검증 방법
	프로그래밍 코드 작성	쉽다
		어렵다
수강동기	적극적	흥미, 향후 활용
	수동적	필수이수, 타인이 수강

양적 연구에서는 학습자가 제출한 프로그래밍 실습 과제물을 채점 결과를 분석하였다. 매주 부여되는 총 10번의 실습에 대한 프로그래밍 제출물이 수집되었으며 실습은 각각 7개의 문제로 구성하였다. 제출된 과제물은 모두 Python 구현 결과로 코드 작성과 실행 결과를 채점하여 10점의 척도로 평가하였다. 성취도 분석에서는 내용 분석에서 얻은 수강 동기의 유형과 계열별 변인으로 비교하여 차이를 비교하였다. 이를 통해 학습자의 심리적 요소와 전공 영역이 프로그래밍 학습의 어려움에 어떤 영향을 나타내는지 살펴보았다.

Ⅳ. 결과 및 분석

4.1 새로운 문제 해결 방식의 어려움

표 2의 분석 결과 비전공 학습자들은 프로그래밍 언어 학습 자체에서 겪는 어려움뿐만 아니라 새로운 학습 분야에서 경험하는 문제들에서도 어려움을 나타내었다. 이는 교수자가 컴퓨팅사고 개념과 프로그래밍 언어 문법과 작성 방법 등의 내용 주로 초점을 두고 교육을 하지만, 비전공 학습자는 익숙하지 않은 도구 사용, 문제의 이해, 풀이 결과의 확인 방법 등에서 혼란을 겪어 수업 내용에 대한 어려움이 가중되는 요인이 된다.



<그림 1> 실습 진행에 따른 일지의 내용 비율
<Fig. 1> Percentage of log information according to categories over laboratories.

그림 1은 일지의 분석 결과에서 새로운 문제 해결 방식의 각 범주에 해당하는 일지 내용의 비율을 학기 진행 시간순으로 나타낸 것이다. 학기 초반에는 새로운 도구 사용에 대한 피드백이 가장 많았으며 특히 Python 쉘(Shell)과 편집기의 용도를 구분하는 데 큰 어려움을 나타냈다. 작성한 소스 코드 제출물 분석에서도 쉘에 출

력된 실행 결과를 답안으로 혼동하여 소스 코드 대신 제출하거나, 쉘에서 작성한 코드와 실행 결과를 제출하거나, 하나의 소스 파일에 여러 문제의 코드를 혼합하여 작성하는 등의 도구 사용의 미숙함과 프로그램 코드와 실행 결과의 차이를 인식하지 못하여 발생한 오류도 있었다. 그 외에도 자신이 저장한 소스 파일의 위치 찾기와 재실행 방법에서도 어려움을 겪었다. 수업 자료와 강의로 이전에 설명된 내용이지만 일부 학습자들은 쉽게 적응하지 못하는 것으로 나타났다.

학기 중반 이후의 실습에서는 도구 사용에 대한 어려움은 줄어들었으나 대신 문제의 요구사항 이해와 올바른 결과 확인 방법에 대한 일지 내용이 증가하였다. 프로그래밍 문제에서 자주 사용되는 정수와 실수의 자료유형 구분, 초과와 이상의 범위 비교, 반올림이나 내림의 근삿값 연산, 대소문자 구분 등과 같은 문제의 요구사항을 이행하지 못하는 경우가 많았다. 예를 들어 '체중을 실수로 입력받으시오' 의 요구사항을 일상생활에서 많이 사용하는 정수만 입력을 받는 것으로 이해하거나, '대소문자를 구분하지 않고 영어 단어를 찾으시오' 에서 'THE' 와 'the' 가 같은 검색 대상이라는 것을 의미하는 문장으로 이해하는데 어려움을 겪는 경우가 많았다.

이와 같은 프로그래밍 문제에서 사용되는 용어나 요구사항에 익숙해지면서 결과물을 검증하는 방식의 어려움이 증가하였는데, 예시로 제시된 일부 입력에 대한 올바른 결과만 나오도록 문제를 풀이하여 예시와 다른 입력에 대해서 올바른 결과가 나오지 않는 경우가 많았다. 학습자들은 다양한 입력에 대해서 올바르게 동작해야 하는 컴퓨팅 문제 해결 방식에 대한 이해가 부족하여 결과 검증 과정을 생략하는 경우가 많았으며, 여러 가지 테스트 케이스를 입력하고 올바른 답안을 도출 하는 문제 해결 방식에 어려움을 표현했다.

4.2 코드 구현 과정의 어려움

코드 구현 과정에서는 표 3과 같이 각 세부 범주가 비슷한 수준의 어려움을 나타내었다. 이전 교육과정에서 다루어온 산술 및 논리 연산은 비교적 쉽게 작성하였으며, 제어 구조의 경우에도 입문자 수준의 문제 해결 과정에서는 큰 어려움을 겪지 않았다. 그러나 학기 후반부의 지역 및 전역 변수의 올바른 사용, 리스트의 변경 연산, 문자열 연산 및 클래스 작성 등의 새로운 개념에서 다소 어려움을 나타냈다.

<표 3> 프로그래밍 요소에 따른 성취도 분석 결과
<Table 3> Academic Achievement Score for the Programming Elements

분류	학습요소	N	평균	표준편차
프로그래밍 코드 구현의 어려움	Input / Output	159	8.22	2.22
	Data Types	159	9.40	1.27
	Conditional Structure	159	9.13	1.34
	Iterative Structure	159	9.64	1.06
	List I	159	9.36	1.06
	List II	159	9.27	1.10
	Function	159	8.37	1.06
	Dictionary	159	8.71	1.47
	String and Text File	159	9.14	1.44
	Module and Class	159	9.09	1.52
전체 평균		159	9.23	0.64

4.2.1 수강 동기에 따른 차이

표 4.는 수강 동기에 따른 어려움의 차이를 분석한 결과이며 적극적 동기를 가진 그룹은 101명, 소극적 동기를 가진 그룹은 58명이다. 모든 주별 점수에서 levene 등분산 검정이 $p>0.05$였으므로 대응표본 T 검정을 실시하여 성취도 차이의 유무를 측정하였다. 그 결과 적극적 동기를 가진 그룹에 비해 수동적 동기를 가진 학습자가 초반 이후에 학습 성취도가 유의미하게 낮아졌으며 문제 해결에서 어려움을 느끼는 정도가 큰 것으로 나타났다.

적극적인 수강 동기의 집단은 입출력, 자료유형, 제어 구조와 같은 기본적인 프로그래밍 언어의 개념부터 이후 후반부의 내용까지 성취도가 크게 변화가 없어 전체적으로 어려움을 덜 느끼는 반면, 소극적 수강 동기의

집단은 자료 유형 이후 논리 구조를 이용한 문제 해결부터 후반부까지 성취도가 떨어지면서 점점 더 학습에 어려움을 느끼는 것으로 나타났다.

<표 4> 수강 동기에 따른 성취도 분석
<Table 4> Academic achievement scores by motivation

학습요소	수강동기	평균	표준편차	t	p
Input & Output	적극적	8.39	2.09	1.295	0.099
	수동적	7.94	2.42	2.34	
Data Types	Active	9.65	0.72	0.359	0.360
	수동적	9.56	1.10	1.61	
Conditional Structure	Active	9.29	1.20	1.986	0.024
	수동적	8.86	1.47	1.54	
Iterative Structure	Active	9.71	1.03	2.347	0.010
	수동적	9.24	1.26	1.22	
List I	Active	9.57	0.86	3.300	0.001
	수동적	9.01	1.26	1.05	
List II	Active	9.48	0.88	3.138	0.001
	수동적	8.93	1.32	1.19	
Function	Active	8.50	0.90	2.242	0.013
	수동적	8.13	1.25	1.13	
Dictionary	Active	8.85	1.55	1.702	0.045
	수동적	8.42	1.36	1.37	
String & Text File	Active	9.41	1.26	3.229	0.001
	수동적	8.68	1.60	1.54	
Module & Class	Active	9.16	1.46	3.306	0.001
	수동적	8.24	2.17	1.69	
Overall	Active	9.19	0.60	4.537	0.000
	Passive	8.66	0.92	0.74	

4.2.2 계열에 따른 차이

계열별로는 자연 및 공학계열과 인문 및 사회 계열을 포함한 나머지 계열들을 비교하여 표 5와 같이 분석하였다. 계열별 결과에서도 levene 등분산 검정이 모두 $p>0.05$였으므로 대응표본 T 검정을 실시하여 성취도 차이의 유무를 측정하였다. 기존 연구에서 자연·공학계열의 학생들이 스크래치 프로그래밍 교육에서 더 잘 적응하고 성취도가 높은 것으로 분석된 결과도 있었으나[19], 본 연구에서 두 집단의 성취도는 전체적으로 유의미한 차이가 없었다.

<표 5> 인문·사회 및 이학·공학계열에 따른 성취도
<Table 5> Academic achievement scores by colleges

학습주제	계열	평균	표준편차	t	p
Input & Output	이학·공학	8.38	1.91	0.701	0.242
	인문·사회	8.13	2.38	2.34	
Data Types	이학·공학	9.65	0.71	0.285	0.388
	인문·사회	9.60	0.96	1.61	
Conditional Structure	이학·공학	9.10	1.35	-0.219	0.414
	인문·사회	9.15	1.36	1.54	
Iterative Structure	이학·공학	9.71	0.96	1.366	0.087
	인문·사회	9.43	1.13	1.22	
List I	이학·공학	9.35	0.93	-0.114	0.455
	인문·사회	9.38	1.13	1.05	
List II	이학·공학	9.38	0.98	0.952	0.171
	인문·사회	9.21	1.16	1.19	
Function	이학·공학	8.33	1.24	-0.350	0.363
	인문·사회	8.39	0.94	1.13	
Dictionary	이학·공학	8.52	1.77	-1.147	0.126
	인문·사회	8.80	1.31	1.37	
String & Text File	이학·공학	9.37	1.16	1.587	0.057
	인문·사회	8.98	1.57	1.54	
Module & Class	이학·공학	9.09	1.48	1.443	0.076
	인문·사회	8.66	1.96	1.69	
Overall	이학·공학	9.06	0.78	0.762	0.223
	인문·사회	8.96	0.77	0.74	

Ⅴ. 결 론

본 연구에서는 비전공자를 대상으로 한 Python 프로그래밍 교육 과목에서의 학습자들의 오류들을 분석하고 다양한 변인으로 분석하였다. 분석 결과를 통하여 나타난 시사점을 정리하면 다음과 같다.

첫째로 대부분의 학습자에게 있어서 프로그래밍은 기존에 접해보지 못한 학습 주제이며 개발 도구의 사용법, 컴퓨팅 문제의 유형, 문제 해결 방식 및 제출 방식에 익숙하지 않기 때문에 학기 초반 시점에 학습 내용에 집중하는데 방해 요소로 작용하고 어려움을 가중시킬 수 있다.

비전공자가 스크래치와 같은 교육용 프로그래밍 언어(Educational Programming Language:EPL)이 아닌 범용 프로그래밍 언어(General-purpose Programming Language;GPL)를 학습하는 과정에서 새로운 프로그래

밍 도구 사용으로 인한 어려움을 줄이는 방법으로는 구글 코랩(Google Colab.)이나 Code.org와 유사한 온라인 개발환경 인터페이스를 제공하여 도구 설치의 생략 및 결과물 작성 및 제출 방법의 자동화로 학습자의 부담을 최소화하고 프로그래밍 개념 학습에 집중할 수 있는 환경을 제공하는 것이 필요하다.

또한 정해진 입력값을 받아 알고리즘에 따라 처리하고 올바른 결괏값을 도출하는 컴퓨팅 문제의 해결 방식을 학기 초기에 이해하고 확립하는 것이 중요하다. 컴퓨팅사고를 통한 문제 해결은 요구사항의 정확한 이해를 하고 해결 과정을 통해 도출된 결과가 요구사항을 만족는지 검증하는 전체 단계가 이루어져야 하나, 학습자들은 프로그래밍 명령어를 사용하여 자신이 작성한 코드가 별다른 문법 오류 없이 결과를 출력되면 문제 해결한 것으로 여기는 경향이 있었다.

컴퓨팅사고의 문제의 정의, 데이터 표현 등의 개념과 올바른 알고리즘 사고를 위해서는 단순히 결과의 정답 여부로 성취도를 측정하는 것이 아니라 문제별로 입력 데이터와 출력 데이터를 구체적으로 정의하고, 테스트 플랜(test plan)을 작성하고 적용하도록 하여 컴퓨팅 문제 해결 방식을 이해하고 올바른 알고리즘을 작성하였는지 검증할 수 있는 역량을 초반부에 확립하는 접근이 필요하다.

다음으로 프로그래밍 언어의 내용 학습에 있어서 이전 교육과정에서 학습한 산술 연산 및 논리 표현에서는 어려움이 없었지만 함수, 자료구조, 문자열 처리, 클래스와 같은 새로운 개념에서 어려움을 나타내는 것으로 나타났다.

수강 동기의 경우 적극적인 동기를 가진 집단이 전체 과정에서 어려움을 덜 겪었으며 심리적 요소가 학습에 영향을 미치는 것을 확인하였다. 반대로 수동적인 동기를 가진 집단은 학습 성취도가 진도에 따라 낮아지는 경향이 있었으며 상대적으로 어려움을 더 경험하는 것으로 나타났다. 최근의 SW과목의 필수화와 함께 프로그래밍에 대해 낮은 동기를 가진 학습자들도 과목을 수강해야 하는 상황이 발생하면서 이러한 두 집단 간의 격차를 최소화하는 것이 필요하게 되었다. 학습 초반부에 흥미와 동기를 유발할 수 있도록 실생활에 적용할 수 있는 문제와 계열별 전공 분야에 활용할 수 있는 구성의 과목 설계가 한 방안이 될 것이다.

그러나 내용 분석에서 학습자들이 일반적으로 가지는 '자연·공학계열 학생들이 프로그래밍을 잘 할 것이다' 라는 학습자들의 인식과는 달리 계열별 변인 비교에서는 집단 간에 유의미한 차이가 없었다. 이는 소속 전공이나 계열보다는 학습 동기와 노력이 중요함을 학습자들에게 주지시키고 이전 학습자들의 사례를 소개하며 자신감을 주는 것이 필요하다.

대학에서 비전공자를 대상으로 하는 프로그래밍 관련 교양 과목은 학습자들이 처음 접하는 문제 해결 방식, 정해진 시수내에 습득해야 하는 생소한 문법과 표현, 필수 이수 요건으로 인한 낮은 동기 부여, 여러 계열의 학생들이 동시에 수강함으로 생기는 자신감 저하 등 여러 가지 문제점들 포함하고 있다. 향후 연구로는 분석된 결과와 제시된 방안을 토대로 하여 어려움을 효과적으로 해결할 수 있는 학습 모델의 설계가 필요하다.

참 고 문 헌

[1] J. Lee and E. Yoon, "The Effects of CS Unplugged Education on the Computational Thinking of Gifted and Talented Students", Journal of Creative Information Culture, vol.6, no.2, pp.77-88, 2020.

[2] Y. Jeon, "Sequence Analysis and Suggestion for Linking Contents of Elementary and Secondary Software Education", Journal of Creative Information Culture, vol.5, no.2, pp.105-116, 2019.

[3] Ministry of Science and ICT, https://www.korea.kr/news/pressReleaseView.do?newsId=156381631 (visited Jan. 2021)

[4] SPRI, "New Education and Talent Strategies Needed in the Era of Digital Transformation", Monthly Software Oriented Society, no.70, 2020

[5] Korea Health Industry Development Institute Breif, "Artificial Intelligence (AI) Manpower Nurturing Policies and Implications by Major Country", Korea Health Industry Development Institute Issue Paper, vol. 276, 2019

[6] Y.H. Kim, J.Y. Yoo and N.J. Kim, "Elementary and secondary software education Operational status and improvement tasks", National Assembly Research Service, vol.34, 2019

[7] E. Kang, S. Shin and K. Lee. "Education Model Using PBL for IT Convergence Education of Non-Major in Liberal Arts Class: Focusing on Computing Thinking", Journal of Digital Contents Society, vol.20, no.11, pp.2159-2168, 2019

[8] J.K. Kim, "Computational Thinking on Problem Solving Process in SW Education for non-CS Major Students", Journal of Korea Multimedia Society, vol.22, no.4, pp.472-479, 2019

[9] L. Gouws, K. Bradshaw and P. Wentworth, "First year student performance in a test for computational thinking", in Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference, New York, USA, pp.271-277, 2013.

[10] S. Lee and S. Ahn, "Influence of Convergence Education Based on Computing Thinking Ability on Problem Solving Ability and Interest", Journal of Creative Information Culture, vol.6, no.3, pp.179-187, 2020.

[11] M. Konecki, "Problems In Programming Education And Means Of Their Improvement", B. Katalinic, Ed, Vienna, Austria:DAAAM International Scientific Book, pp.459-470, 2014

[12] Y. Bosse and M. Gerosa, "Why is programming so difficult to learn?: Patterns of Difficulties Related to Programming Learning Mid-Stage", ACM SIGSOFT Software Engineering Notes, vol.41, no.6, pp.1-6, 2016

[13] C. Lee, "Elementary School Teachers' Difficulties in Learning Programming EPL", Journal of Korean Practical Arts Education, vol.32, no.2, pp.49-63, 2019

[14] S. Kim, "Analysis of Non-Computer Majors' Difficulties in Computational Thinking Education", The Journal of Korean Association of Computer Education, vol.18, no.3, pp.49-57, 2015

[15] J. Sung, S. Kim and H. Kim, "Analysis of Art and Humanity Major Learners' Features in Programming Class", The Journal of Korean Association of Computer Education, vol.18, no.3, pp.25-35, 2015

[16] J. Choi and Y. Lee, "The analysis of Learners' difficulties in programming Learning", The Journal of Korean Association of Computer Education, vol.17, no.5, pp.89-98, 2014

[17] J.W. Creswell, Qualitative, Quantitative and Mixed Method Approach, 3rd Ed., LA:SAGE Publications, 2009

[18] Y. Sung, "Mixed Methodologies of Qualitative and Quantitative Research for Education Research", Journal of Elementary Education, vol.30, no.2, 2014

[19] M. Oh and M. Kim, "Analysis of Effects of Scratch Programing Education to Improve Computational Thinking", Korean Association for Educational Information and Media, vol.24, no.2, pp.255-275, 2018

저 자 소 개

김 재 경(Jaekyung Kim) 정회원

■2007년 2월 : 연세대학교 컴퓨터과학과 (공학박사)
■2009년 6월 : 연세대학교 컴퓨터과학과 연구교수
■2014년 9월 : 연세대학교 컴퓨터과학과 객원교수
■2017년 3월 ~ 현재 : 연세대학교 학부대학 조교수

<관심분야> : SW교육, 데이터과학
e-mail : kim.jk@yonsei.ac.kr

손의성(Eisung Sohn) 정회원

■2012년 8월 : 연세대학교 컴퓨터과학과 (공학박사)
■2012년 8월 : 연세대학교 소프트웨어응용연구소 연구원
■2018년 3월 ~ 현재 : 연세대학교 학부대학 조교수

<관심분야> : SW교육, 컴퓨터그래픽스
e-mail : esohn@yonsei.ac.kr