JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 20, No. 11, pp. 2159-2168, Nov. 2019

교양수업에서 비전공자의 IT 융합 교육을 위한 PBL 활용 교육 모델-컴퓨팅적 사고 중심으로

강 의 선[1*] · 신 선 임[1] · 이 광 진[2]
[1]숭실대학교 베어드교양대학
[2]중앙대학교 유럽문화학부

Education Model Using PBL for IT Convergence Education of Non-Major in Liberal Arts Class: Focusing on Computing Thinking

Eui-Sun Kang[1*] · Sun-Im Shin[1] · Kwang-Jin Lee[2]

[1]Baird College of General Education, Soongsil University, Seoul 06978,Korea

[2]School of European Languages and Cultures, Chungang University, Seoul 06974, Korea

[요 약]

최근 ICT 기반의 디지털 융합 시대를 주도할 인재를 육성하기 위해 대학에서는 컴퓨팅 사고력을 함양하기 위한 교과과정들을 생성 및 운영하고 있다. 컴퓨팅 사고력은 실세계 문제를 컴퓨터가 효과적으로 수행할 수 있도록 해결안에 대한 일련의 과정들을 절차적, 체계적으로 기술하는 사고 과정이다. 하지만 컴퓨팅 사고력 함양을 위한 소프트웨어 교육이 비전공자에게는 생소한 개발 환경, 전공과의 관련성 등으로 인해 난관에 부딪히고 있다. 이를 해결하기 위하여 본 논문에서는 비전공자를 위한 교육용 프로그래밍 언어를 이용하여 소프트웨어 교육에 활용하였고 PBL을 활용하여 전공과 관련된 어플리케이션 및 아이디어를 컴퓨팅적 사고의 구성요소와 논리적 흐름으로 분석을 하도록 하였다. 본 연구는 교양 수업에서 비전공자들에게 소프트웨어의 개념 뿐만 아니라 컴퓨팅 사고력과 전공과의 융합에 대한 이해를 향상시킬 수 있는 수업 모델의 예를 제시해 줄 수 있을 것이다.

[Abstract]

Recently, In order to bring up talents who will lead the era of digital convergence based on ICT, universities are creating and operating curricula to build computational thinking. Computational thinking is a process of thinking that describes a set of processes procedurally and systematically so that computers can effectively solve real-world problems. However, Software education for developing computing thinking is facing difficulties due to the unfamiliar development environment and relevance to the major. In order to solve this problem, in this paper, the educational programming language for non-specialists was used for software education, and the applications or idea related to the major were analyzed in terms of the components and logical flows of the commercial applications. This study can provide non-majors with examples of instructional models that can improve not only the concept of software but also the understanding of computing thinking and convergence with majors.

색인어 : 컴퓨팅적 사고, 비전공자, PBL, 문제해결, 소프트웨어 교육
Key word : Computational Thingking, Non-Computer Major, PBL, Problem Solving, Software Education

http://dx.doi.org/10.9728/dcs.2019.20.11.2159

Received 01 October 2019; **Revised** 22 October 2019
Accepted 05 November 2019

***Corresponding Author; Euisun Kang**

Tel: +82-2-828-7264
E-mail: iami86@ssu.ac.kr

Ⅰ. 서 론

2016년 세계경제포럼에서 화두된 4차 산업혁명은 인공지능(AI), 사물 인터넷(IoT) 등의 ICT 기술을 이용한 초연결화(hyperconnectivity)와 초지능화(Superintelligence)를 기반으로 기존 산업과의 융·복합 중요성을 강조하면서 산업, 경제, 사회 분야에 대한 미래의 그림을 제시하고 있다. 이런 4차 산업혁명의 변화의 시작에는 기술 간의 융합을 가능하도록 도와주는 소프트웨어(Software)의 확산에 있다. 디지털 시대는 물리적인 영역과 사이버 영역을 연결하는 소프트웨어에서 벗어나기 힘들 뿐만 아니라 모든 영역에서의 소프트웨어와의 융합은 더욱 가속화 될 것이다. 이를 위한 준비로써 현실 세계의 문제를 컴퓨터 과학 원리를 적용하여 해결할 수 있는 컴퓨팅적 사고 능력의 필요성을 강조하고 있다[1]. 컴퓨팅적 사고는 현실 세계의 문제를 정보처리 관점에서 문제를 분석하고 해결 과정을 설계 및 평가할 수 있는 융복합 사고력을 의미한다[1][13]. 2018년도부터 초, 중, 고, 대학에서는 소프트웨어 교육을 통하여 컴퓨팅 사고력을 함양하기 위한 교육을 진행하고 있고 그에 대한 연구가 활발히 이루어지고 있다[2]. 소프트웨어 교육은 컴퓨팅 사고력을 함양하는 목표도 있지만 소프트웨어적 문제 해결 역량을 높여 비전공자가 소프트웨어 전문가와 협업할 수 있는 의사소통 능력을 높이는데도 있다. 하지만 비전공자의 경우 정보처리 관점의 절차적 문제해결과정이 익숙하지 않을 뿐만 아니라 프로그램 작성 환경 및 경험 부재, 전공과의 관련성 등으로 학습의 어려움들이 보여지고 있다[15][16]. 본 논문은 비전공 학습자들을 대상으로 단순한 프로그래밍 언어 및 소프트웨어에 대한 지식 전달 뿐만 아니라, 학습자의 전공지식 및 일반적인 상황에 적용할 수 있는 컴퓨팅 사고력을 함양하는데 그 목적이 있다. 이를 위해 본 논문에서는 소프트웨어의 다양한 경험을 위해서 비전공자의 특성을 고려한 블록 기반인 스크래치와 텍스트 기반의 파이썬을 학습한다. 또한 소프트웨어 교육의 단점을 보완하기 위하여 소속 학과의 전공과 융합하여 개발된 상용 어플리케이션 또는 IT 제품 사례를 찾고 분석하여 컴퓨팅적 사고의 구성요소 및 논리적 흐름으로 분석하도록 하였다. 이를 위한 과정은 PBL 방식을 활용하였다. 이에 대한 효과성을 확인하기 위해 S대학의 교양필수 교과목인 “컴퓨팅적 사고”에 적용하였다. 이에 대한 연구 결과로써 사후 설문 조사 결과와 PBL의 결과물들을 제시하고자 한다.

Ⅱ. 관련연구

2-1 컴퓨팅적 사고

2006년 Wing[3]은 일상의 문제를 해결하는 과정에서 문제를 분석하여 문제의 핵심요소를 추출하고 이를 정형화하여 해답을 찾아 컴퓨터로 자동화 할 수 있는 능력을 컴퓨팅 사고라 정의하였다. 즉, 실세계의 문제를 컴퓨터 기기로 자동화하기 위해서는 현실 문제를 정확히 이해하고 개념화하여 인간의 상상력, 창의력, 사고력을 컴퓨터에 심어주어 컴퓨터가 사람처럼 행동할 수 있도록 하는 것이다. 컴퓨팅 사고력은 컴퓨터 전공자뿐만 아니라 모든 분야에서도 적용 가능한 보편적인 사고로써, 읽기, 쓰기, 셈하기와 같은 근본적인 학습이 되어야 한다고 주장하고 있다. 컴퓨팅 사고력을 함양하기 위해 학습해야 할 중심 모델로써, 영국의 BBC에서는 컴퓨팅적 사고를 분해, 패턴인식, 추상화, 알고리즘의 4가지 구성요소를 제안하고 있다[4]. 미국의 컴퓨터과학교사협회(CSTA)와 국제교육공학협회(ISTE)는 학생들이 문제를 해결하는 과정을 자료수집, 데이터 분석, 데이터 표현, 문제 분할, 추상화, 알고리즘 및 절차, 자동화, 시뮬레이션, 병렬화의 9단계 과정으로 분리하여 문제를 정의하고 해결방안을 모색할 수 있도록 가르치고 있다[5][6]. 2015년 한국교육개발원은 컴퓨팅 사고의 학습 모델로써 분해(Decomposition), 패턴인식(Pattern Recognition), 추상화(Abstraction), 알고리즘(Algorithm)의 4가지와 프로그래밍(Programming) 선택적 항목을 포함한 DPAA(P)을 소프트웨어 교수-학습 모델로써 제시하였다[6].

2-2 소프트웨어 교육 현황 및 교육 방법

컴퓨팅적 사고력이 강조되면서 국내에서는 2015년도 교육과정 개정을 통해 2018년부터 소프트웨어 교육을 교과과정에 포함함으로써 소프트웨어 교육의 필요성을 강조하였다[7]. 미래창조과학부와 정보통신기술진흥센터는 4차 산업혁명의 디지털 시대를 이끌어갈 창의적 인재를 육성하기 위한 일환으로 소프트웨어 중점대학 사업을 시작하였다. 고려대를 포함한 8개 대학에서 시작한 소프트웨어 중점사업은 2019년까지 35개 대학으로 확대하여 시행되고 있다.

컴퓨팅 사고력을 함양하기 위한 소프트웨어 교육 방법으로는 언플러그드(Unplugged), 피지컬 컴퓨팅(Pysical computing), 블록기반 프로그래밍(EPL; Educaitonal Programming Language), 고급 프로그래밍 언어 교육이 있다[6]. 언플러그드 교육은 학생들이 컴퓨터 없이 퍼즐, 게임 또는 카드 놀이등 다양한 도구를 활용하여 컴퓨터에 대해 이해하는 과정을 학습하는 방법이다[8]. 피지컬 컴퓨팅은 프로그래밍 학습을 위해 블록, 회로 및 로봇등의 센서를 활용하는 방법이다[9]. 블록 기반 프로그래밍은 전공자가 프로그래밍 개발을 위해 사용하는 텍스트 기반의 고급 프로그래밍이 아닌 레고(Lego)와 같은 블록 쌓기 방식으로 프로그램의 개념 및 흐름을 학습할 수 있는 방식이다[10]-[12]. 고급 프로그래밍 언어 교육은 대학에서 많이 사용하는 교육으로써 파이썬과 같은 텍스트 기반의 코딩 교육이다[9].

2-3 교양 교과에서 비전공자 대상의 소프트웨어 교육에 대한 고찰

최근 대학들은 학생들의 창의적 사고력을 소프트웨어로 구

현 가능한 인재를 발굴하기 위한 소프트웨어 중점대학의 목표를 기반으로 IT 전공자 외에 비전공자에게도 소프트웨어 교육을 의미화하여 교양 교과과정을 편성하고 있다. 하지만 컴퓨팅 사고력 함양만을 볼 때 문화적, 사회적 주제의 전체적인 문제 분석을 중점으로 학습한 비이공계열 학생들에는 절차적 문제해결 중심인 컴퓨팅 사고 교육이 생소함을 느끼는 요인으로 보여지고 있다[13]. 이 문제를 해결하기 위하여 많은 연구에서 컴퓨터의 이해 및 컴퓨팅 사고력 증진을 위하여 실습을 통한 프로그래밍 교육의 필요성을 언급하고 비전공자 학습자를 위한 코딩교육이 제시되고 있다. 컴퓨팅 사고력의 이론수업을 실제 코딩 실습수업으로 연계하면서 학습자는 정보처리기기로 자동화하기 위한 문제 분석 과정 및 알고리즘 기술 과정을 통하여 문제 해결 역량 및 논리적 사고, 창의력을 키울 수 있다. 또한 프로그래밍 실습은 자연스럽게 컴퓨팅적 사고력에 대한 학습자의 내적 동기와 지적 호기심을 향상 시킬 수 있음을 지적하였다[14]. [9]는 비전공자를 대상으로 실시한 프로그래밍 학습의 체감 난이도가 이론 수업에서는 전공자와 차이가 없으나 실습, 과제, 팀 프로젝트와 같은 실제 프로그램을 구현하는 측면에서는 체감 난이도가 높음을 확인하였다. [15]는 스크래치를 이용한 프로그래밍 교육에서 코딩 오류에 대한 수정, 전공과의 비관련성, 문법의 어려움과 프로그래밍의 생소함에 의해 프로그래밍 학습을 어려워한다고 하였다. [16]은 비전공 학습자들은 변수, 리스트의 개념, 아이디어를 생각하고 프로그램으로 작성하는 과정에서 어떤 명령어를 선택해야 하는지에 대한 고민 순으로 어려움이 있음을 확인하였다.

2-4 문제 해결 중심의 교육 방법

비전공자들이 컴퓨팅 사고력을 함양하기 위해서는 소프트웨어 교육 중심이든 컴퓨팅 사고력 중심이든 교수자가 아닌 학생 스스로 문제를 해결할 수 있는 능력을 배양하는 것이 중요하다. 기존의 컴퓨팅 사고력 함양은 정형화된 강의계획에 의한 교수 중심의 수동적 수업이거나 프로그래밍 실습과 예제 중심의 수업이 주를 이루고 있다[15][16]. 컴퓨팅 사고력의 이해와 학습자의 내적, 지적 호기심을 향상시키기 위해 소프트웨어 교육 중심으로 학습이 이루어지는 것은 옳으나 고등학교에서 사회적, 문화적인 전반적인 문제를 기반으로 고등학교에서 학습된 비전공자인 경우 일반적인 사고력과 현실 문제를 컴퓨터의 세계에서 자동화하여 옮기는 사고력의 관계는 낯설게 느껴질 수 있다[16]. 이를 위해서 학생 스스로 문제를 발견하고 구성원과의 상호작용에 의해 지식을 습득할 수 있는 문제중심학습 또는 프로젝트 중심 학습을 고려해 볼 필요가 있다. PBL(Problem Based Learning)은 1994년 Barrows에 의해 소개된 학생 중심의 교육 설계 방법으로써 학습자의 긍정적인 학습 참여를 통해 자기 주도적인 학습 능력을 기르는 데 효과가 있음을 증명하였다. Barrows는 문제기반 학습의 특징으로 학습이 소집단 안에서 학생중심으로 이루어지며 교육자는 조력자, 안내자의 역할을 한다. 또한 문제해결 능력을 개발하는 수단으로서의 문제는 학습을 위한 자극과 핵심으로써 조직한다고 하였다. Barrows는 PBL의 단계를 수업 전개 단계, 문제 제시 단계, 문제 후속 단계, 결과물 제시 및 발표 단계, 문제 완결과 해결이후의 단계로 구성하였다[17][18]. 이를 논문[19]는 PBL 교수 설계 모형으로 분석, 설계, 개발, 구현, 평가의 3가지 디자인 모델로 분류하고 5개의 PBL을 컴퓨터 공학 입문 수업에 적용하여 운영하였다. 그 결과 학습 내용에 대한 이해, 협동학습에 대한 이해, 실제적 경험, 창의 문제 해결력, 프리젠테이션 스킬, 의사소통 능력, 자기 주도적 학습 능력, 자신감과 같은 긍정적인 효과를 확인하였다. 그 외에도 학생 중심의 문제 해결 방법으로는 인간 중심의 문제를 도출하고 단계적, 체계적으로 문제를 해결해나가는 디자인 씽킹[20], 창의적 문제 해결 능력을 갖춘 현장 실무형 인재양성을 위한 캡스톤디자인[21] 등이 있고 그 효과성을 각 연구에서 입증하였다.

비전공자를 대상으로 창의력 사고와 문제해결능력을 향상시킬 수 있는 전략은 소프트웨어 중심 교육에서 벗어나 PBL과 디자인 싱킹과 같은 교육과정을 활용해야 한다. 하지만 기존 PBL을 통한 문제해결방식은 블록기반 방식의 스크래치나 앱인벤터를 이용한 PBL활용 방법이 중심을 이룬다. 비전공자를 위한 소프트웨어 교육은 기초수준의 교육이다[24]. 따라서 주제의 범위 및 복잡도에 따라 프로그래밍의 난이도가 구현 범위를 벗어날 수 있으며 이에 따른 결과가 학생들에게 프로그램에 대한 거부감을 줄 수 있다. 뿐만 아니라 그룹 시간 조율, 참여 시간 등에 대한 어려움을 표현하기도 한다[19]. 이를 위해 본 논문에서는 전공 분야 및 응용 가능한 어플리케이션을 컴퓨팅적 사고의 구성요소로 분석함으로써 소프트웨어 개발 과정 및 논리적 흐름의 중요성을 간접적으로 경험 할 수 있도록 하였다. 이 과정에서 비전공자의 논리적 사고력과 문제 해결 능력 향상을 위해 문제중심학습인 PBL 방식을 활용하고자 한다.

Ⅲ. 교양수업에서 비전공자의 컴퓨팅 사고력 함양을 위한 PBL 적용 사례

컴퓨팅 사고력은 일반적인 문제를 이해하고 분석하여 정형화하는 과정과 문제의 해결방안을 찾는데 관련된 사고의 과정이다. 과정의 결과는 정보처리기기에 의해 실행될 수 있는 형태로 표현된다[1]. 이런 창의적 사고를 계발하기 위한 효율적인 교육법을 모색하고 효과성을 확인하고자 본 논문에는 S대학교에서는 2017년 신설된 교양필수 교과목인 “컴퓨팅적 사고”에 적용하였다. 이 교과목의 교육목표는 다음과 같다.

첫째, 프로그래밍 경험을 통해 4차 산업혁명의 중심인 소프트웨어를 이해한다.

둘째, 문제를 이해하고 문제해결과정을 논리적이고 체계적으로 표현하는 능력을 계발한다.

셋째, 자신의 전공에 소프트웨어 기술 및 컴퓨터 과학을 융합할 수 있는 시야와 사고력을 계발한다.

이 교과목은 비전공자인 인문, 사회, 경상, 법학, 예체능계열

학생들에게 진행된다. 하지만 절차적 문제 해결 중심인 소프트웨어 교육이 생소한 비전공자에게는 컴퓨팅 사고력에 대한 학습자의 내적 동기와 지적 호기심을 유발할 수 있는 방안이 필요하다. 따라서 비전공자들을 대상으로 단순한 코딩지식 전달이 아닌 학습자의 전공지식을 활용할 수 있도록 문제 해결 중심의 PBL 교육 방법을 적용하였다. 이로써 세 번째 교육목표인 학과 전공과 컴퓨터 과학을 융합할 수 있는 시야와 사고력을 계발하고자 한다. 또한 PBL을 통해 문제를 발견하고 해결할 수 있는 문제 해결 역량을 협업 활동을 통하여 증진 시키며 대인관계역량과 커뮤니케이션 역량을 증진시킬 수 있도록 한다. 이로써 미래 직업사회에 적응 할 수 있는 인재를 양성할 수 있도록 하고자 한다.

3-1 교육내용

수업은 매주 온라인 50분, 오프라인 50분의 블렌디드 런닝(Blended Learning)으로 이루어진다. 온라인에서는 컴퓨팅적 사고의 정의, 컴퓨터, 소프트웨어와 프로그램에 대한 이해 그리고 컴퓨팅적 사고의 구성 요소들에 대한 설명과 예제를 통하여 컴퓨팅적 사고에 대한 이론적 학습을 진행한다. 오프라인은 소프트웨어와 정보처리 기기의 동작 원리에 대한 이해를 돕기 위하여 교육용 프로그램인 스크래치(6주)와 파이션(7주)의 실습을 통해 구체적으로 학습한다. 단, 비전공자라는 점을 감안하여 소프트웨어 개발을 전문적인 단계까지 학습하지는 않는다. 전반적인 주차별 학습내용은 표1과 같다. 오프라인 수업은 매 주차마다 실습 예제가 주어지며 스스로 해결할 수 있는 간략한 문제를 제시하여 과제 게시판에 업로드 하도록 하였다. 이는 추후 평가항목 중 과제 및 참여도(10%)점수로 적용함으로써 모든 학생들이 수업 중 프로그래밍을 통하여 소프트웨어를 이용한 문제 해결 능력을 함양하고자 하는데 있다.

3-2 PBL 진행방법

본 과목은 학생들이 컴퓨터 전공자들이 문제를 분석하고 해결하는 과정을 학습하여 자신의 전공 및 업무 분야에 적용할 수 있는 사고력을 습득하는데 있다. 이를 위해 이론을 통하여 컴퓨팅적 사고력에 대한 기본적인 지식을 습득하고 프로그래밍 언어 중 스크래치와 파이썬을 학습함으로써 디지털 세대에 맞게 소프트웨어 Maker로써의 컴퓨터를 활용할 수 있는 능력을 함양하는데 있다. 따라서 본 연구에서는 학습주체인 학습자가 스스로 문제점을 찾고 그에 대한 해결방안을 협업 활동을 통해 심도있게 사고할 수 있는 기회를 부여하고자 PBL(Project/Problem Based Learning) 교육 방식에서 문제 제시 단계, 문제 후속 단계, 결과물 제시 및 발표 단계 활용하였다[17]. 하지만 소프트웨어 교육을 통하여 학생 스스로 문제를 해결하는 과정을 학습하고 결과물로 PBL을 이용하여 프로그램을 개발하는 단계 대신 분석 결과를 발표하고 보고서를 작성하는 단계로 제한하여 프로그램 개발까지는 진행하지 않았다.

표 1. 비전공자를 위한 컴퓨팅적 사고의 주차별 내용
Table 1. Computational thinking Course syllabus for Non-Major

week	On-Line Content (theory lecture)	Off-Line Content (programming)
1	Lecture Introduction	Understanding and Installing Scratch
2	Understanding Computing Thinking	Scratch behavior, event handling
3	Understanding Software	Form and Observation in Scratch
4	Understanding Programming Language	Scratch Operations and Variables
5	Problem Analysis and Expression 1	Handling Conditions in Scratch Iterating through Scratch
6	Problem Analysis and Expression 2	Dealing with functions in Scratch
7	Decomposition of the Problem 1	Understanding and Installing Python
8	Decomposition of the Problem 2	Python data representation and processing
9	Pattern Recognition 1	Python data representation and processing
10	Pattern Recognition 2	Handling conditions in python
11	Abstracttion 1	Iterating through Python 1 Project presentation,
12	Abstracttion 2	Iterating through Python 2
13	Algorithm 1	Dealing with functions in Python
14	Algorithm 2	Project presentation
15	Algorithm 3	Project presentation

그 이유는 첫째, 이 수업은 교양필수 교과목으로 학생들에게 프로그래밍에 대한 이해와 흥미를 유발하는 초기 단계로써 쉬운 예제 중심으로 실습을 진행하였고 이에 대한 응용문제를 제시하여 간략히 문제를 해결하도록 진행한다. 둘째, PBL 기반의 프로그래밍 교육은 학습자가 선택한 주제의 범위 및 복잡도에 따라 프로그래밍의 난이도가 비전공자가 수행할 수 있는 구현 범위를 벗어날 수 있다. 셋째, 문제를 해결하기 위한 컴퓨터와 프로그램의 복잡한 기술이 학생들에게 프로그램에 대한 거부감을 줄 수 있다. 또한 학생들에게 이 교과목이 소프트웨어를 개발하는 교과목으로 오해를 불러일으킬 수 있다. 따라서 본 교과목에서는 PBL을 활용한 수업 진행 방향으로 컴퓨팅적 사고가 전공이나 일상적인 문제를 해결하는데 도움이 되는지에 대해 초점을 맞추고 연구를 진행하였다.

교양과목에서 비전공자 컴퓨팅 사고력 교육과정의 목표를 달성하기 위한 PBL의 활용 목표는 다음과 같다.

(1) 소통과 협업 과정을 통하여 주제 선정 및 해결과정과 같은 복잡한 문제를 해결할 수 있다.
규모가 크거나 복잡한 문제는 여러 사람이 공유할 경우 업무 분장과 원활한 의사 소통에 따라 결과물의 품질이 달라진다[22]. 주제를 선정하는 과정에서 자료 조사와

공유가 필요하다. 그리고 선택된 주제의 범위에 따라 각자의 역할이 형성된다. 이 과정에서 복잡한 문제에 대한 소통과, 협업에 대한 중요성을 배울 수 있다.

(2) 문제(주제)에 대한 해결과정을 절차적, 체계적, 논리적으로 표현할 수 있다.
주어진 문제 또는 주제에 대한 해답을 찾기 위한 과정으로 가장 먼저 문제를 분석한다. 문제를 분석한다는 것은 문제에 대한 정확한 이해를 의미한다. 이런 분석을 구체적인 요소로 분리하고 세분화 할 수 있다면 논리를 바탕으로 절차적이고 체계적으로 기술할 수 있다. 또한 해결과정을 명시적이고 시각적으로 표현한다면 해결방법을 구성하고 검토할 수 있는 역량을 계발할 수 있다[23].

(3) 컴퓨팅 사고력을 현실 세계의 문제 또는 전공과 연계할 수 있는 융합적 시야와 사고력을 계발한다.
비전공자의 입장에서는 컴퓨팅 사고력이 전공자만을 위한 필수 요소라 생각할 수 있다. 하지만 전공과 관련된 IT 제품을 검색하고 분석하는 과정에서 융합을 통한 전공의 기여도 및 활용 범위를 다시 생각해 볼 수 있는 기회를 제공하고 소프트웨어 교육을 통해 실세계와 컴퓨터 환경의 차이를 이해 할 수 있다. 이런 학습 과정은 소프트웨어 전문가와의 협업과정에서 발생할 수 있는 분야간 차이를 극복하는데 도움을 줄 것이다.

조별 프로젝트는 3~4명이 한 팀을 구성하도록 하였다. 주제는 전공과 IT를 결합할 수 있는 제품이나 아이디어라면 무엇이든지 가능하며 주제가 난해하다면 실생활에 필요한 IT 결합 아이디어이면 선택하도록 하였다. 그 결과로써 회의록, 발표 자료, 발표 영상, 최종 보고서, 성찰 일지를 제출하도록 하였다. 보고서 및 발표 내용은 조에서 선택한 주제를 문제로 인식하고 문제해결과정에 학습한 컴퓨팅적 사고의 구성요소(자료수집, 분석, 분해, 추상화(일반화), 패턴인식, 알고리즘)를 활용하여 기술하도록 하였다. 이런 결과물들은 학습자들에게 컴퓨팅적 사고에 대한 이론적인 내용을 기반으로 전공 및 실생활에 문제해결과정으로 적용해 볼 수 있는 기회를 제공할 수 있다.

1) 자료 수집 및 주제 선정 단계

이 단계에서는 협업을 통하여 문제를 찾고 조별 주제를 선택하는 단계이다. 주제 선정을 위한 다양한 자료를 수집함으로써 문제의 난이도 및 다양한 방법을 찾을 수 있는 계기를 마련하도록 하였다.

3~4명이 한 조를 이루어 전공과 IT를 결합할 수 있는 제품, 상품 또는 캠퍼스나 세상에 영향을 줄 수 있는 실제 문제들을 선정하도록 하였다. 만약, 전공과 연관지어 생각하기 어렵다면 일상생활에서 IT를 결합할 수 있는 아이디어들을 제시하도록 하였다. 문제 선택 후 문제의 범위를 재설정하기 위하여 사전에 교수자와 문제 범위 조정에 대한 논의를 통하여 학습자가 프로젝트 수업에 부담이 없도록 진행하였다.

표 2. 컴퓨팅 사고력의 DPAA(P) 모델[6]
Table 2. DPAAP Model of Computational Thinking[6]

Components	Content
Decomposition	Breaking down of a system into smaller parts that are easier to understand, program and maintain.
Pattern Recognition	Recognition Searching or identifying patterns and regularities in data
Abstraction	Simplification and formalizing found principles with pattern recognition
Algorithm	A plan, a set of step-by-step instructions to solve a problem

2) 문제 해결 과정 단계

이 단계는 컴퓨팅 사고에서 제시하는 구성요소를 바탕으로 주어진 문제를 논리적으로 분석하고 이를 글, 기호 또는 그림으로 해결 과정을 구체적으로 기술하는 단계이다. 가장 먼저 주제를 바탕으로 문제를 분석하여 문제를 정확히 이해하도록 한다. 그리고 컴퓨팅적 사고의 구성요소를 이용하여 분석한 자료를 구체화하고 과정을 기술하도록 하였다. 컴퓨팅 사고의 구성요소는 2015년 한국교육개발원의 5단계 학습 모델[6]을 본 교과목의 환경과 특성에 맞게 분해, 패턴인식, 추상화, 알고리즘으로 분리하여 작성하도록 하였다. 이 모델 중 자동화를 위한 프로그래밍은 제외하였다. 선택된 4가지 구성요소에 대한 정의는 표 2와 같다. 각 조에서 결정된 주제에 대해서 컴퓨팅적 사고에서 다루어지는 4가지 구성요소를 활용하여 문제를 분석하고 해결하도록 하였다.

① 분해하기
큰 문제를 작은 문제들로 나눠 문제를 세분화하여 기술하고 동시에 역할 분담이 가능하도록 한다.
② 패턴 찾기
문제에서 반복적인 사항이나 규칙들을 찾아 기술한다.
③ 추상화(일반화)하기
복잡한 문제를 단순화하여 문제를 일반화 시키고 세부사항들을 분리하여 구체적으로 기술한다.
④ 알고리즘 작성하기
각각의 작은 문제들의 흐름을 잘 정의된 절차적 방법으로 단계적이고 체계적으로 기술한다. 기술 방법은 알고리즘 표현 방법인 자연어 또는 순서도를 이용하여 작성하도록 하였다. 그리고 분해 과정에서 만들어진 작은 문제들의 흐름을 기반으로 전반적인 절차를 기술한다.

3) 조별 결과물 작성단계

선정된 주제를 중심으로 분석된 문제 해결 과정 단계의 각 구성요소들을 요약하고 기술하는 단계이다. 조별 결과물로는 회의록, 최종 보고서, 발표자료, 성찰일지가 있다.

① 회의록
원활한 협업 활동을 진행하기 위하여 회의 시간, 회의 장소, 회의 참여자 명단, 회의 주제 및 내용이 포함된 회의록을 작성하도록 하였다.

② 최종 보고서(조별)
문제제시, 문제 분석 결과, 문제해결과정의 각 단계에 대한 결과를 보고서 형식으로 작성 후 과제 게시판에 제출하도록 하였다.
③ 발표 자료 및 영상
발표하는 모습을 영상으로 촬영(10분이내)하여 과제 게시판에 업로드 하도록 하였다. 조별 발표 유형은 비전공자의 편의를 위하여 단순 발표, 역할극 및 토론 등 자유롭게 선택하도록 하였다. 제출 가능한 자료는 PPT, 대본 또는 그 외 자료이다. 이는 짧은 수업시간을 감안하고 평가 기준을 바탕으로 교수자 외에 학생들이 직접 평가에 참여하기 위함이다.
④ 성찰 일지
프로젝트를 진행하는 과정에서 느낀 어려움, 교과목의 이해 정도, 컴퓨팅적 사고의 이해 정도 등을 파악하기 위해 개별적으로 자유롭게 기술하도록 하였다.

Ⅳ. 연구 결과

본 교과목은 컴퓨팅적 사고에 대한 이론과 프로그래밍 실습을 통하여 비전공자에게 컴퓨터 분야에서 문제를 해결하는 사고방법을 전달하고 각 전공에서 활용 및 응용할 수 있도록 하고자 하는데 있다. 수업에서 이루어지는 이론 및 실습은 학생들이 주입식 교육으로 받아들여질 수 있으므로 조별 프로젝트 과정을 통해 팀별로 학습자의 전공과 연관된 IT 기술들을 조사하고 선택된 특정 제품 및 아이디어를 컴퓨팅적 사고를 활용하여 문제해결과정으로 분석하고 적용해 보는 시간을 가져 보았다. 그리고 사후 설문을 통하여 본 연구에 대한 효과성을 알아보았다.

그림 1. 발표 자료의 예
Fig. 1. Presentation Example

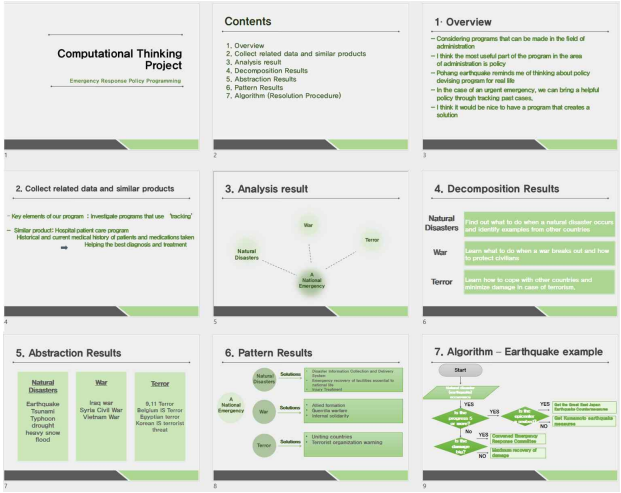


표 3. 응답자 성별 분포
Table 3. Respondent gender ratio

Gender	Distribution	Ratio
Fale	98	41.7%
Female	137	58.2%
Total	235	100%

표 4. 응답자 소속 대학 분포도
Table 4. College profile of respondents

Gender	Distribution	Ratio
Humanities	90	38.1%
Business	77	32.6%
Social Science	47	19.9%
Economics and Commerce	21	8.9%
Total	235	100%

표 5. 프로그래밍 언어의 흥미도
Table 5. Programming language interest rate

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	9.1%	9.1%	39.0%	35.1%	7.8%
Economics &Commerce	4.8%	19.0%	19.0%	47.6%	9.5%
Social Science	8.5%	12.8%	31.9%	40.4%	6.4%
Humanities	17.8%	12.2%	31.1%	27.8%	11.1%
Total	10.0%	13.3%	30.3%	37.7%	8.7%

4-1 제출한 발표 영상 및 발표 자료

발표영상은 발표 시간문제로 모두 발표하지 못할 경우를 대비하여 발표하는 모습을 영상으로 촬영(10분 내외)하여 과제 게시판에 제출하도록 하였다. 발표자의 영상노출 의무는 부여하지 않았다. 하지만 몇몇 조는 단순한 발표 모습 대신에 문제해결과정을 대화형식으로 풀어나가는 역할극 형식으로 촬영하기도 하였다.

4-2 설문 결과

본 교과목을 마친 후 학생들의 학업 성취도 및 컴퓨팅 사고력의 이해도를 파악하기 위해 사후 설문을 실시하였다. 본 설문에서는 235명이 응답하였다. 표 3과 표 4에서 볼 수 있듯이 응답자의 성별 분포에서는 남학생보다 여학생의 비율이 약 17% 많았다. 그중 38.1%로 인문대학 학생들이 가장 많았고 다음으로 경영대학 학생들이 많았다.

1) 프로그래밍에 대한 흥미도

비전공 학생들에게 프로그래밍은 생소할 수 있다. 본 논문은 실습 과정의 프로그래밍에 대한 흥미도를 확인해 보았다. 표 5에서 보이는 바와 같이 프로그래밍 흥미도는 '그렇다' 이상의 응답이 46% 였다. 하지만 표 6의 프로그래밍 경험 유무에서 불

수 있듯이 이미지 기반의 스크래치는 이전 경험이 약 35%정도 있었고 텍스트 기반의 파이썬은 약 8%로 경험이 없는 경우가 대부분이었다. 전체 프로그래밍 흥미도의 응답은 이미 이전 경험이 있는 학생들이 갖는 흥미도에 영향을 받을 수 있지만 전반적으로 긍정적인 효과를 가졌다고 할 수 있다.

또한 표 7에서 볼수 있듯이 쉬운 블록 기반의 스크래치에 비해 파이썬에 많은 흥미를 갖고 있음을 알 수 있다. 프로그램의 경험 유무와 비교해 봤을 때 이는 대학 수준에 맞는 새로운 언어에 대한 지적 호기심에 의한 결과라 할 수 있다.

표 6. 프로그래밍 경험 유무
Table 6. Programming experience

	Scratch		Python	
	Distribution	Ratio	Distribution	Ratio
Strong Neg.	112	47.7%	162	68.9%
Neg.	24	10.2%	33	14.0%
Normal	14	6.0%	19	8.1%
Pos.	51	21.7%	13	5.5%
Strong Pos.	34	14.5%	8	3.4%

표 7. 프로그래밍 종류에 따른 흥미도
Table 7. Interest rate by type of programming

	Distribution	Ratio
Scratch	111	47.2%
Python	124	52.8%

표 8. 컴퓨팅적 사고를 이용한 문제해결 효율성 정도
Table 8. Efficiency degree of problem solving using computational thinking

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	6.5%	9.1%	45.5%	32.5%	6.5%
Economics &Commerce	4.8%	14.3%	47.6%	28.6%	4.8%
Social Science	14.9%	12.8%	40.4%	29.8%	2.1%
Humanities	14.4%	20.0%	44.4%	17.8%	3.3%
Total	10.1%	14.0%	44.5%	27.2%	4.2%

표 9. 전공이나 일상적인 문제 해결을 위한 활용도
Table 9. Utilization to solve major or general problems

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	1.3%	6.5%	39.0%	36.4%	16.9%
Economics &Commerce	0.0%	19.0%	9.5%	47.6%	23.8%
Social Science	8.5%	10.6%	21.3%	46.8%	12.8%
Humanities	13.3%	8.9%	33.3%	28.9%	15.6%
Total	5.8%	11.3%	25.8%	39.9%	17.3%

2) 컴퓨팅 사고력의 문제 해결 효율성

개인의 성향이나 학습방법 및 과정에 따라 문제의 해결방법이 다르게 나타날 수 있다. 교과 내용을 통해 문제를 해결하는 것이 전공이나 개인적인 성향에 비추어 새로운 사고력을 함양하는데 도움이 되는지 확인해 보았다. 표 8의 컴퓨팅 사고력을 이용한 문제해결 효율성 부분에서 31%만이 긍정적인 효과가 나타남을 확인할 수 있었다. 이는 교양 교과에 주어지는 부담과 사고력은 짧은 시간에 함양하는 데 무리가 있음을 알려주는 결과라 할 수 있다. 하지만 컴퓨팅적 사고를 통한 다른 분야의 적용도를 확인해 보았을 때 표 9의 결과에서 볼 수 있듯이 약 56% 정도가 전공이나 일상적인 문제 해결을 위해 새로운 사고를 활용할 의지를 가지고 있음을 확인할 수 있었다. 특히 데이터를 다루어야 하는 경제통상대학의 경우 학과 전공의 학습내용과 진로에 영향을 미칠 수 있으므로 활용 측면에 대해 높은 결과를 보였다.

3) 프로젝트를 통한 전공과 IT 융합의 이해 정도

이 교과목에서는 컴퓨팅적 사고에 대한 이론적인 내용 뿐만 아니라 프로그래밍 실습을 통한 소프트웨어 교육까지 함께 이루어지고 있다. 이 과정에서 비전공자들은 IT 계열 교과목에 대한 거부감을 느끼고 있었고 이를 전공과의 융합으로 이끌어내는 것이 쉽지 않았다[15][16]. 이를 극복하고자 전공과 IT의 융합이라는 주제로 전공과 관련된 상용 어플리케이션 또는 전공과 연결할 수 있는 아이디어를 컴퓨팅적 사고 요소를 기반으로 분석하는 시간을 가졌고 이 과정에서 PBL을 활용하여 진행하였다. 이로써 본 교과목의 목표에 달성하고자 시도하였다. 그 결과 표 10에서 볼 수 있듯이 전공과 IT 융합에 대해서는 교과목 이해도가 52%였다. 이로써 전공과 연결할 수 있는 융합교육이 문제 해결력, 사고력 향상 효과와 관련이 있음을 확인할 수 있었다[14].

표 10. 전공과 IT과의 융합에 대한 이해 정도
Table 10. Understanding of convergence between major and IT

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	3.9%	7.8%	36.4%	37.7%	14.3%
Economics &Commerce	0.0%	14.3%	23.8%	33.3%	28.6%
Social Science	8.5%	8.5%	25.5%	42.6%	14.9%
Humanities	11.1%	12.2%	34.4%	28.9%	13.3%
Total	5.9%	10.7%	30.0%	35.6%	17.8%

표 11. 프로젝트를 통한 교과목 이해의 정도
Table 11. Degree of understanding of the subject through the project

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	2.6%	10.4%	39.0%	37.7%	10.4%
Economics &Commerce	4.8%	14.3%	14.3%	42.9%	23.8%
Social Science	8.5%	4.3%	17.0%	63.8%	6.4%
Humanities	4.4%	7.8%	32.2%	46.7%	8.9%
Total	5.1%	9.2%	25.6%	47.8%	12.4%

표 12. 프로젝트를 통한 컴퓨팅적 사고에 대한 이해 정도
Table 12. Understanding of computational thinking using project

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	1.3%	5.2%	36.4%	37.7%	19.5%
Economics &Commerce	0.0%	19.0%	4.8%	57.1%	19.0%
Social Science	8.5%	6.4%	14.9%	63.8%	6.4%
Humanities	6.7%	5.6%	32.2%	47.8%	7.8%
Total	4.1%	9.0%	22.1%	51.6%	13.2%

표 13. 알고리즘 기술 방법에 대한 효과성
Table 13. Effectiveness of Algorithm Expression Method

	Strong Neg.	Neg.	Normal	Pos.	Strong Pos.
Business	3.9%	7.8%	44.2%	33.8%	10.4%
Economics &Commerce	4.8%	14.3%	23.8%	38.1%	19.0%
Social Science	6.4%	14.9%	31.9%	44.7%	2.1%
Humanities	12.2%	14.4%	37.8%	28.9%	6.7%
Total	6.8%	12.9%	34.4%	36.4%	9.6%

4) 컴퓨팅적 사고의 이해를 위한 PBL 활용의 효과성

전공 어플리케이션 분석 및 아이디어를 구성하는 과정에서 PBL을 활용한 프로젝트를 통하여 교과목과 컴퓨팅 사고의 이해 정도에서 표 11와 표 12과 같이 각각 60%와 64%의 긍정적인 답변을 확인하였다. 하지만 문제해결과정을 알고리즘으로 표현하는 효과성에 대해서는 높은 긍정적인 효과를 볼 수 없었다. 이는 컴퓨터 과학에서 사용하는 문제해결 기술 방법이 문장 중심에 익숙한 학생들에게는 여전히 생소하게 받아들여질 수 있다[2]. 따라서 프로젝트 주제를 좀 더 명확히 할 필요와 프로젝트 진행 기간을 늘릴 필요가 있고 실습 역시 일상적인 생활에 깊이 연관 지을 수 있는 예제로 재편성할 필요가 있다[19].

4-3 성찰일지에 작성된 주관식 응답

설문 결과와 성찰일지를 보면 타 전공에 비해 인문계열 학생들이 전공과 IT를 결합할 수 있는 아이디어를 도출해 내는 것은 어려워보였다. 이는 학습자의 전공 관련성에도 영향이 있음을 알 수 있다. 하지만 컴퓨팅적 사고력을 활용한다는 점에 대해 새로운 도전에 관심을 보였고 그 결과 IT와 전공의 융합부분에 대해 긍정적인 반응을 보였다.

응답1) 문제의 패턴을 찾는 것과 추상화라는 것의 개념이 가장 어려웠던 것 같다.
응답2) 문제분석, 패턴, 추상화와 같은 컴퓨터의 여러 가지 개념을 전공과 관련시키려고 하는 과정에서 정확한 개념을 파악하기 위해 수업했던 내용을 다시 찾아보는 복습의 계기도 되었던 것 같다.
응답3) 전혀 컴퓨터와 관련 없을 것이라 생각한 우리의 전공이 프로그램으로 연결될 수 있다고 생각하니 새로운 경험이었다.
응답4) 이번 활동을 통해 인문학을 공부한다고 해서 인문학만 고집하는 편협한 시각보다는 보다 넓은 시각으로 여러 분야를 두루 살필 수 있는 계기가 된 것 같다.
응답5) 평소에 불편을 겪고 있었던 문제를 해결할 수 있는 프로그램을 생각해 볼 수 있었다는 점에서 앞으로 컴퓨팅 기술이 나의 전공과 관련된 일에도 충분히 활용될 수 있다는 점을 느꼈다.
응답6) 우리가 구상한 이 프로그램이 현실화된다면 나와 같은 고고학을 어려워하는 사람도 좀 더 쉽게 고고학을 공부할 수 있지 않을까 하는 생각이 들었다.
응답7) 막상 프로그램을 생각하다보니 행정과 IT의 결합은 필수적이라는 것을 알 수 있었다. 컴퓨팅적 사고의 접근이 실생활에 얼마나 유용한지 깨닫게 되는 유익한 시간이었다.
응답8) 프로젝트를 준비하면서 아이디어나 상황을 설정하는 것도 그렇고 준비해야 할 것들이 많아 약간은 고단한 과제라는 생각이 들기도 했다. 하지만 직접 이렇게 준비하고 해나가는 과정을 거쳐서 '컴퓨팅적 사고를 통한 문제 해결 과정'을 확실히 내 머릿속에 남긴 것 같다. '컴퓨팅적 사고' 수업을 계기로 일상생활을 하면서 IT와 연관해볼 수 있는 것들에 대해 많이 생각하게 될 것 같다.
응답9) 문제를 파악하고 방법을 통해 문제 해결 방식을 도출하는 순서의 큰 틀은 비슷하지만, 그 속에서 서로 다른 현상에서 비슷함을 찾는 패턴 분석, 복잡한 문제를 간단히 나타내는 추상화, 사고의 과정을 도식화하는 알고리즘 등은 힘든 와중 우리에게 신선함을 주었다.

IV. 결론 및 향후과제

ICT 기술을 이용한 4차 산업혁명은 소프트웨어의 발전에 힘입어 산업 간의 융합을 더욱 활성화하고 경제, 산업, 사회 등 모든 분야에 많은 미치는 영향을 줄 것이다. 4차 산업혁명을 살아가는 디지털 네이티브 세대는 이런 변화 속에서 갖추어야 할 다음 세대에 안내자 역할을 할 것이다. 이를 위해 필요한 컴퓨팅 사고력은 실세계의 문제를 정보과학의 관점에서 창의적이고 논리적으로 접근하고 해결방안을 모색해야 하지만 ICT 기술에 의한 산업 간의 융합을 위해 타인과의 협업을 통해 최적의 해결안을 찾아내는 의사소통 능력이 필요하다. 이를 위해 2019년도에는 소프트웨어 육성사업을 35개 학교로 확대하고 컴퓨팅 사고력 함양 및 소프트웨어 교육을 강화하고자 노력하고 있다. 하지만 비전공자들에게 소프트웨어 교육은 흥미도, 자신감, 숙련도, 지적 호기심 정도에 따라 다른 결과를 보이고

있으며 컴퓨터 과학과의 융합에 대한 필요성을 크게 느끼지 못하고 있다[9][15][16].

본 논문에서는 이런 문제점을 극복하는 방법으로써 ICT 기술 확산의 중심에 있는 소프트웨어 및 컴퓨터에 대한 이해를 돕기 위해 교육용 프로그램 실습을 실시하였다. 또한 PBL을 활용하여 전공 분야의 상용 어플리케이션이나 전공 분야를 활용한 아이디어를 컴퓨팅적 사고의 구성요소를 활용하여 흐름을 기술하는 방법을 적용하여 보았다. 그 결과 PBL 교육 방법이 교과목에 대한 이해와 컴퓨팅 사고에 대한 이해에 60%이상 도움이 되는 것을 확인할 수 있었다. 이는 전공이나 일상적인 문제 해결을 위한 활용도에도 영향을 미치는 것을 알 수 있었다. 하지만 전공과 IT 융합을 위한 어플리케이션 분석과 아이디어를 도출하는 프로젝트 수업이 소프트웨어 이해에 미치는 영향에 대한 향후 연구가 필요하며 PBL을 활용하여 아이디어를 프로그램 개발까지 완료하는 계속적인 연구를 통하여 효과성을 입증하는 연구가 진행되어야 할 것이다. 또한 학과 전공의 특성과 관심도에 따라 적절한 소프트웨어 교육의 학습 모델과 이에 따른 체계적인 PBL 교수법에 관한 추가적인 연구가 진행되어야 할 것이다.

참고문헌

[1] S. j. Ahn, K. S. Noh, K.S. Oh, *Computational Thinking*, EhanMeida, 2019.

[2] M. J. 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, 2017

[3] Wing, J. M., “Computational Thinking,” *Communications of the ACM*, Vol. 49, No. 3, pp. 33–35, 2006.

[4] BBC. Introduction to computational thinking. BBC Bitesize. https://www.bbc.com/bitesize/guides/zp92mp3/revision/1.

[5] CSTA & ISTE, Computational thjnking in K-12 education teacher resource second edition, CSTA & ISTE: https://www.iste.org/docs/ct-documents/ct-teacher-resources_2ed-pdf.pdf?sfvrsn=2/.

[6] H. So. Kang, J. H. Cho, H. C. Kim, “A Study on Software Analysis and Design Education Model based on Computational Thinking,” *Journal of Digital Contents Society*, Vol. 20, No. 5, pp. 947-954, May 2019.

[7] https://www.moe.go.kr/.

[8] https://csunplugged.org/en/.

[9] J. Seo, “A Case Study on Programming Learning of Non-SW Majors for SW Convergence Education,” *Journal of Digital Convergence*, Vol. 15, No. 7, pp. 123-132, 2017.

[10] https://scratch.mit.edu/

[11] https://playentry.org/.

[12] https://appinventor.mit.edu/.

[13] J. Y. Park, “Direction and prospect of SW education in the 2015 revised curriculum,” *KEDI Journal of educational policy*, Vol. 42, No. 3, pp. 85–89, 2015.

[14] S. Y. Pi, “A Study on Coding Education of Non-Computer Majors for IT Convergence Education,” *Journal of Digital Convergence*, Vol. 14, No. 10, pp. 1-8, 2016.

[15] M. J. Oh, “Non-Major Students’ Perceptions of Programming Education Using the Scratch Programming Language”, *The Journal of Korean Association of Computer Education*, Vol. 20, No. 1, pp. 1-11, 2017.

[16] S. H. Kim, “Analysis of Non-Computer Majors’ Difficulties in Computational Thinking Education”, *The Journal of Korean Association of Computer Education*, Vol. 18, No. 3, pp. 15-23, 2015.

[17] Barrows, H. S. & Myers, A.: Problem-based learning in secondary schools. Unpublished monograph, Springfield. IL: Problem-based learning Institute, Lanphier School, and Southern Illinois University Medical School. 1994.

[18] I. S. Park, “A Study Basic Engineering for Improving the Creative Practice PBL Case,” *Journal of the Korea Academia-Industrial cooperation Society*, Vol. 14, No. 11, pp. 5396-5402, 2013.

[19] K. S. Lee, “Study of Applications of Intro sduction of Computer Engineering Class using PBL,” *Journal of the Korea Academia-Industrial cooperation Society*, Vol. 15, No. 10, pp. 6303-6309, 2014

[20] E. K. Suh. “Development of Creative Thinking and Coding Course method on Design Thinking using Flipped Learning,” *Journal of Learner-Centered Curriculum and Instruction*, Vol. 17, No. 16, pp. 173-199, 2017.

[21] H. S. Kang, J. H. Cho, H. C. Kim, “Case Study on Capstone Design Model for Computer Engineering,” *Journal of Digital Convergence*, Vol. 14, No. 5, pp. 57-66, 2016.

[22] Niko M., Roman B., Erkki S., Mordechai Ben-Ari, “Extending the Engagement Taxonomy : Software Visualization and Collaborative Learning”, *Proceedings of the 37th SIGCSE technical symposium on Computer science education*, Vol. 9, No. 1, pp. 1–27, 2009.

[23] K. Charntaweekhun, S. Wangsiripitak, “Visual Programming using Flowchart”, *International Symposium on Communications and Information Technologies*, pp. 1062-1065, 2006

[24] G. J. Park, Y. J. Choi, “Exploratory study on the direction of software education for the non-major undergraduate students,” *Journal of Education & Culture*, Vol. 24, No. 4, pp. 273-292, 2018.

강의선(Eui-Sun Kang)

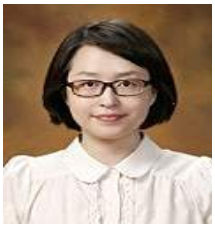
2002년 : 숭실대학교 (공학석사)
2007년 : 숭실대학교 (공학박사-미디어공학)

2007년~현 재: 숭실대학교 베어드교양대학 부교수
※관심분야 : 멀티미디어(Multimedia), 사물 인터넷(Internet of Things), 컴퓨터 교육(Computer Education) 등

신선임(Sun-Im Shin)

2006년 : Washington State University (교육학석사-상담심리 전공)
2013년 : 서울대학교 대학원 (철학박사-교육상담 전공)

2013년~현 재: 숭실대학교 베어드교양대학 조교수
※관심분야 : 상담(Counseling), 수퍼비전(Supervision), 대인관계(relationship), 진로(career), 학교생활 적응 (academic adoptation) 등



이광진(Kwang-Jin Lee)

2003년 : 중앙대학교 대학원 (문학석사)
2009년 : Paris VIII 대학교 대학원 (문학박사-현대문학)

2012년~2018년: 숭실대학교 베어드교양대학 조교수
2019년~현 재: 중앙대학교 유럽문화학부 부교수
※관심분야 : 문학교육(education de literature), 프랑스문학(french literature), 정신분석학(psychoanalysis) 등