Journal of the Korea Convergence Society
Vol. 9. No. 3, pp. 73-78, 2018

ISSN 2233-4890
https://doi.org/10.15207/JKCS.2018.9.3.073

컴퓨터 비전공자를 위한 파이썬 기반 소프트웨어 교육 모델

이영석
강남대학교 KNU 참인재대학 교양교수부

Python-based Software Education Model for Non-Computer Majors

Youngseok Lee
KNU College of Liberal Arts and Sciences, Kangnam University

요 약 컴퓨팅 기술을 다양한 분야와 융합하여 새로운 가치를 만들어내고자 하는 노력이 현대 사회에서 강조되고 있다. 이제 소프트웨어를 설계하고 제작하는 능력을 포함한 컴퓨터 소양 교육은 전공분야와 상관없이 누구에게나 이뤄져야 하는 사회 보편적인 교육으로 자리 잡고 있다. 많은 대학들이 컴퓨터 비전공 학생들을 포함하여 컴퓨팅 기술을 활용한 문제 해결력을 향상시키기 위해 소프트웨어 교육을 필수 이수하도록 시도하고 있다. 하지만, 아직은 컴퓨터 전공 학생들을 위한 프로그래밍 교육 관점에서의 소프트웨어 교육을 실시하다 보니 프로그래밍 언어 문법을 학습하는 과정에서 많은 어려움을 호소하고 있다. 이러한 문제를 해결하기 위하여, 본 논문에서는 기존의 소프트웨어 교육 모델 연구결과를 분석한 뒤, 컴퓨터 비전공자를 위한 파이썬 기반 소프트웨어 교육 모델을 제안한다. 이를 위해, 파이썬 기반 소프트웨어 교육 모델을 위한 학습 절차와 교수 전략 및 한 학기 분량의 커리큘럼을 제안하였으며, 교양 수업에 적용하여 유의미한 결과를 도출하였다. 제안하는 소프트웨어 교육 모델을 적용한 강의가 진행한다면 학생들에게 흥미와 관심을 유도하면서 컴퓨팅 사고력과 문제 해결력을 향상시킬 수 있을 것이다.

주제어 : 소프트웨어 교육, 교육용 프로그래밍 언어, 파이썬 교육 모델, 교양 교육, 컴퓨터 프로그래밍

Abstract Modern society has evolved to such an extent that computing technology has become an integral part of various fields, creating new and superior value to society. Education on computer literacy, including the ability to design and build software, is now becoming a universal education that must be acquired by everyone, regardless of the field of study. Many universities are imparting software education to students to improve their problem-solving ability, including to students who are not majoring in computers. However, software education contains courses that are meant for computer majors and many students encounter difficulty in learning the grammar of programming language. To solve this problem, this paper analyzes the research outcomes of the existing software education model and proposes a Python-based software education model for students who are not majoring in computer science. Along with a Python-based software education model, this paper proposed a curriculum that can be applied during one semester, including learning procedures, and teaching strategies. This curriculum was applied to a liberal arts class and a meaningful result was derived. If the proposed software education model is applied, the students will be interested in the computer literacy class and improve their computational thinking and problem-solving ability.

Key Words : Software Education, Educational Programming Language, Python Education Model, Liberal Arts Education, Computer Programming

*This Research was Supported by Kangnam University Research Grants.(2016)
*Corresponding Author : Youngseok Lee (yslee38@kangnam.ac.kr)
Received January 29, 2018 Revised March 2, 2018
Accepted March 20, 2018 Published March 28, 2018

1. 서론

다가오는 4차 산업혁명 시대에는 소프트웨어가 가치 창출과 혁신의 중심이 될 것으로 예상되고 있다. 이에 우리나라에서도 '소프트웨어 중심 사회를 위한 인재양성 추진 계획'에서 제시한 바와 같이, 국가 차원의 정보 교육 과정을 발표하면서 소프트웨어 교육을 강조하고 있다[1]. 일례로, 많은 대학들이 전공지식과 함께 소프트웨어를 직접 설계하고 제작할 수 있도록 소프트웨어 교육을 전공 분야와 상관없이 모든 학생들에게 실시하고 있다[2].

이러한, 소프트웨어 교육의 방향은 컴퓨팅 사고(Computational Thinking)를 바탕으로 문제 해결을 위한 아이디어를 얻고, 자신만의 결과물을 만드는 과정을 통하여 창의력을 배양하도록 하고 있다[3, 4]. 컴퓨팅 사고란 컴퓨터 과학의 기본 개념과 원리를 토대로 한 문제 인식 및 문제 해결력을 말하며 미래 인재가 기본적으로 갖추어야 할 소양이라고 할 수 있다[4, 5].

소프트웨어 교육을 통하여 컴퓨터 비전공자들은 자신의 전공 분야에서 컴퓨팅 기술을 활용하여 자신의 아이디어를 구체화 할 수 있고, 다양한 분야의 전문가와 소통하는 능력을 갖추도록 도울 수 있다. 이를 위한 컴퓨팅 사고력은 컴퓨터 기반의 사고를 해야 하므로, 교육용 프로그래밍 언어 기반의 소프트웨어 교육을 통해서 개발하고 향상시킬 수 있다[5].

소프트웨어 교육을 위한 교육용 프로그래밍 언어의 종류는 다양하지만, 컴퓨터 비전공자 학생들에게 관심과 흥미를 가지게 하면서 학습하기 쉽고, 다양한 형태의 응용 프로그램으로 확장할 수 있는 언어로 접근하는 것이 바람직하다[5]. 파이썬 프로그래밍 언어는 비교적 쉽게 배울 수 있고 그래픽 처리 기능이 단순하기 때문에 초보자가 처음 배우기에 적절하다[6].

또한 다양한 앱이나 웹 형태로 개발하기도 유용하므로 융합형 교육을 위한 프로그래밍 언어로도 활용 가능성이 높다[6]. 특히 미국 내에서 인지도가 높은 컴퓨터 전공이 개설된 대학 중 상위 69%가 파이썬을 채택하여 교육하고 있다[7]. 이렇듯 국내외 현황을 살펴보면, 파이썬은 대학에서 컴퓨터 비전공자에게 교육용 프로그래밍 언어로 적합하다고 판단할 수 있다.

대부분의 대학에서 컴퓨터 비전공자를 위한 소프트웨어 교육은 비전공자를 위한 별도의 교육 목표와 방법 등을 충분히 고민하지 않고, 컴퓨터 전공 학생들을 위한 프로그래밍 언어 중심의 교육을 그대로 옮겨 실시하고 있어 컴퓨터 비전공 학생들이 소프트웨어 교육 과정에서 학습의 많은 어려움을 호소하고 있다[4, 5].

따라서 본 논문에서는 기존의 소프트웨어 교육 모델을 분석하고 컴퓨터 비전공자들이 쉽고 재미있게 배울 수 있는 파이썬 기반 소프트웨어 교육 모델과 학습 절차, 커리큘럼을 제안하여 교양 교육에 적용하고자 한다.

2. 관련 연구

2.1 소프트웨어 교육 모델

소프트웨어 교육을 통해 학생들의 컴퓨팅 사고력을 향상시킬 수 있는 수업 형태와 수업 모델 관련 연구가 활발히 이루어지고 있으며, 프로젝트 기반 교육, 문제 해결 중심 교육, 짝 프로그래밍, 디자인 사고 기법 등 다양한 방법 등이 대표적이다[8, 9].

한국교육학술정보원에서는 'SW교육 교수학습 모형 개발 연구'를 통해 소프트웨어 교육을 위한 교수학습 모델을 5가지 형태로 제안하고 있다[10].

첫째, CT요소중심(DPAA(P)) 모델은 분해, 패턴인식, 추상화, 알고리즘, 프로그래밍의 5단계를 제시하고 문제해결학습방법과 관련해서 결과에 도달하기까지의 과정을 강조한 모델이다[10].

둘째, 시연중심(DMM) 모델은 시연(Demonstration), 모방(Modeling), 제작(Making)의 단계로 이루어져 있고, 교사의 지도 하에 계속 연습하는 직접 교수법을 바탕으로 한다[10].

셋째, 재구성중심(UMC)모델은 놀이(Use), 수정(Modify), 재구성(reCreate) 단계를 가지고 발견학습법을 바탕으로 하여 학생들이 스스로 학습할 수 있도록 교사가 여건을 제공해 주는 것이 특징이다[10].

넷째, 개발중심(DDD)모델은 탐구(Discovery), 설계(Design), 개발(Development)의 단계로 교사에게 배운 것을 바탕으로 새로운 결과물을 생산하는 형태이다[10].

마지막으로 디자인중심(NDIS)모델은 요구분석(Needs), 디자인(Design), 구현(Implementation), 공유(Share)의 단계로 이루어져 있고, 프로젝트 중심의 수업 형태로서 학생들이 스스로 문제를 인식하고 해결해 나가면서 교사의 조력자 역할을 강조하는 모델이다[10]. 이러한 모델들은 소프트웨어 교육을 진행하는 형태에 따라서 적합한

형태를 취해야 하며, 컴퓨팅 사고력의 구성요소를 단순화하여 실제 수업에 적용할 수 있는 방향을 제시하고 있다[10].

2.2 컴퓨터 비전공자의 소프트웨어 교육

대학에서 프로그래밍 중심의 소프트웨어 교육이 컴퓨터 비전공자들에게까지 확대 실시되면서 컴퓨터 비전공자들에게 효과적인 소프트웨어 교육 방안에 관련된 연구가 활발하게 이루어지고 있다.

서주영(2017)의 연구에서는 비전공자 프로그래밍 학습에 관한 사례 연구를 실시하여 인문 계열과 이공 계열의 차이가 프로그래밍 학습 능력과 느끼는 어려움의 연관은 없지만, 체감 난이도의 차이는 확인하였다[11]. 김수환(2015)의 연구에서는 비전공자 대상의 컴퓨팅 사고 교육에서 변수, 리스트 등 명령어의 사용과 아이디어를 생각하고 구현하는 과정에 대한 어려움을 분석하였다[12].

성영훈(2017)은 컴퓨팅 사고력 개념을 바탕으로 CT 요소와 프레임워크를 분석하면서 학습 전략에 대한 모델링을 실시하여 이를 적용한 결과 HVC(History, VR Coding, Collaboration) 학습 전략이 컴퓨팅 사고력 향상에 유의미하게 나타난 것으로 파악되었다[13]. 박성희(2016)의 컴퓨팅 사고력 함양을 위한 대학에서의 소프트웨어교육에 관한 고찰 연구에서는 컴퓨팅 사고력 향상을 위한 교수학습 모형을 연구하고, 프로젝트학습 모형을 제안하였다[14].

기존 연구들을 분석한 결과, 컴퓨팅 사고력과 소프트웨어 교육에 관련된 연구가 다양하게 진행되고 있지만, 활용 가능성이 높은 파이썬 프로그래밍 교육은 컴퓨터 비전공 학생들에게는 텍스트 코딩 언어로서의 진입장벽으로 인해 운영의 어려움이 있음을 확인하였다.

본 논문에서는 기존 연구들의 분석과 함께 다양한 전공의 대학생들에게 2년 동안 실시해 온 파이썬 기반 소프트웨어 교육 경험을 토대로 컴퓨팅 사고력을 향상시킬 수 있으면서 컴퓨터 비전공자도 쉽게 학습할 수 있는 소프트웨어 교육 모델을 제시하여 적용하고, 이에 대한 효과를 분석하고자 한다.

3. 파이썬 기반 소프트웨어 교육 모델

3.1 소프트웨어 교육 모델 개발 절차

대학 교양 과목에서 소프트웨어 교육을 실시하면서 적합한 교육 모델을 제시하기 위한 분석 절차는 Fig. 1과 같다. 기존의 소프트웨어 교육 관련된 교수학습 모형 개발 연구에서 제시한 모델들과 적용 사례 들을 분석한 결과, 가장 중요하게 고려해야 할 사항이 컴퓨팅 사고력과 생각을 디자인 하는 것이었다.

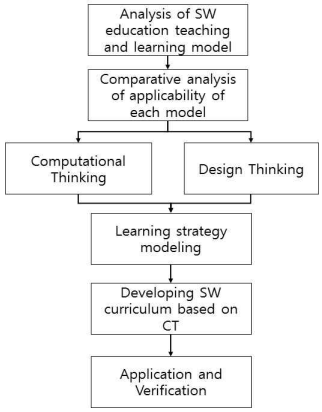

Fig. 1. Analysis procedure for software educational model

컴퓨터 비전공자들은 소프트웨어 교육을 받는 것을 영어를 배우는 것보다 어려워하기 때문에, 쉽고 재미있는 주제나 내용을 바탕으로 컴퓨팅 사고력을 향상시키고 생각하는 것을 표현하는 방법이 필요하다. 따라서 기존 모델의 적용 가능성을 판단하고, 그에 따른 학습 전략을 수립하고, 적합한 콘텐츠와 주제를 개발하여 실제 적용해 보고 그 가능성을 파악해 보고자 하였다.

3.2 소프트웨어 교육 모델 학습 전략

소프트웨어 교육 모델 개발 절차에 따라서 제안한 학습 전략은 Fig. 2와 같다.

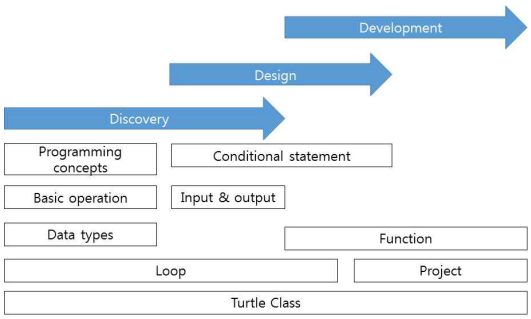

Fig. 2. Learning progression of SW educational model

컴퓨팅 사고력 향상을 위한 컴퓨터 프로그래밍 교육의 영향 연구에서 볼 수 있듯이, 교육용 프로그래밍 언어의 일반적인 주제와 콘텐츠에 따라서 강의를 진행했을 때에도 컴퓨팅 사고력이 향상되는 것을 알 수 있다[15]. 이 때 추출된 학습 주제와 콘텐츠와 강의 진행방식은 소프트웨어 교육 교수학습 모형의 개발중심(DDD)모델과 유사했으므로, 이를 활용하기 위하여 단계에 맞도록 교육 요소를 배치하여 학습 전략을 수립하였다[14, 15].

3.3 파이썬 기반 소프트웨어 교육 모델

소프트웨어 교육 모델의 형태와 학습 전략 수립에 따라 컴퓨팅 사고력 핵심 요소와 개발중심(DDD) 모델을 연계하여 비전공자를 위한 파이썬기반 소프트웨어 교육 모델을 제시한 결과는 Table 1과 같다[14, 15].

Table 1. DDD model for non-computer majors

Steps	Key learning method	CT elements
Discovery	Construct knowledge through search and discovery	Pattern
Design	Algorithm planning and design	Abstraction Algorithm
Development	Implementation and feedback in programming languages	Automation Inference

제시한 모델을 실제 강의에 적용하기 위하여 필요한 주제와 내용, 그리고 학습 요소를 제시한 결과는 Table 2와 같다.

Table 2. Python curriculum for SW education

Week	Subject and contents	Learning elements
1	Concept of Pattern, Abstraction, Algorithm, Automation Perform learner diagnostic test, LightBots Game	Understanding CT, programming, learner's level
2	Calculation using variables, drawing various shapes	Numbers, arithmetic expressions, and variables
3	Enter name, age, etc., and process it.	I / O Functions Drawing with the Turtle function
4	Data type, comments, string processing	String type, Boolean, comparison
5	Odd or even judgment	If~else
6	Judging leap year Grading calculation	If~elif~else
7	Generate and calculate random numbers Coin Throw game	list, list index Drawing with the Turtle function
9	Get the sum	For statement
10	Print multiplication	While statement
11	Numbers Matching Game	Complexity loops
12	Find a prime number and divisor	Function
13	Drawing circles, various shapes and coloring	Drawing with the Turtle function
14	Writing a calculator program to calculate arithmetic Turtle Game Projects	Using Functions, Personal Projects

컴퓨터 비전공 학생들의 관심과 흥미를 유도하기 위하여 매주 거북이 클래스를 활용하여 그림 그리기를 실시하면서 프로그래밍의 개념과 기본 연산자, 변수, 자료형 등에 대한 개념을 가르친다. 기본적인 지식에 대한 이해가 되었으면, 본인의 생각을 디자인하기 위해서 입출력과 반복문, 조건문을 배우고, 조건에 따라 다양한 그림을 그리면서 홀수 짝수 판단, 윤년 판단, 성적 처리, 합계 구하기 등의 예제를 코딩해 보는 경험을 갖도록 하였다. 조건문과 반복문의 활용이 원활해지면 실제 개발단계로 들어가면서 함수를 배우고, 함수를 활용하여 게임이나 다양한 그림을 그리면서 자신만의 프로젝트를 완성하도록 하였다.

4. 교육 모델 적용 및 결과 분석

4.1 검사 도구 및 대상 집단

제안하는 소프트웨어 교육 모델이 적합한지 타당성 검토를 하기 위해서 소프트웨어 교육을 실시 후, 학생들의 컴퓨팅 사고력을 측정하기 위해 한국교육학술정보원에서 제시한 컴퓨팅 사고력 검사지를 번안하여 검사도구로 사용하였다[16].

2016학년도 2학기에 K대학교 1학년을 대상으로 Python 언어 기반의 소프트웨어 교육을 실시하였으며 대상 학생에 대한 정보는 Table 3과 같다.

Table 3. Status of participants

Department	Control group	Experimental group
Social Welfare	31	41
Business Administration	28	26
Public Service	21	18
Real Estate	17	15
Economics and Tax Administration	19	16
Total	116	116

2016학년도 2학기 수강생 가운데 4개 분반 약 120명 학생 중 시스템 결측값으로 나타난 4명을 제외하고, 사회복지학부 학생 72명, 경영학부 54명, 행정학과 39명, 부동산 학과 32명, 경제세무학과 35명을 대상 학생으로 하여 각 집단별 동질성 검증을 실시하여 집단간 차이가 없음을 확인하였다.

4.2 집단 특성에 따른 결과

집단 특성에 대한 사전·사후 독립표본 검정을 실시한 결과는 Table 4와 같다.

Table 4. Group characteristics test result (p<.05)

Elements		N	Avg.	Std.	t	P
Grade	Pre	116	1.17	.608	.110	.913
	Post	116	1.16	.589		
Gender	Pre	116	1.47	.501	-.131	.896
	Post	116	1.47	.501		
Class	Pre	116	2.65	1.274	.000	1.000
	Post	116	2.65	1.274		
Time	Pre	116	1.52	.502	.000	1.000
	Post	116	1.52	.502		

분석한 주요요소는 수강 학년(Grade), 성별(Gender), 분반에 따른 차이(Class), 주·야간여부(Time)이고, 컴퓨팅 사고력 측정 요소는 규칙성(Pattern), 추상화(Abstraction), 알고리즘(Algorithm), 자동화(Automation), 추론(Inference), 컴퓨팅 사고력 전체(Total)로 구분하였다. 사전 검사에 비해서 사후 검사의 문항 수준이 좀 더 어려움에도 불구하고 평균 수치는 큰 차이가 없음을 알 수 있다.

독립표본 검정을 실시한 결과를 보면, 수강하는 학생 대부분이 신입생이어서 학년에는 유의미한 차이가 나타나지 않는다. 특이한 것은 성별, 다른 분반, 주야간에 따라서도 유의미한 차이가 나타나지 않는 것이다. 이는 학기 초에 학생들 진단을 통해서 동질성 검증이 완료된 상태에서 강의가 진행되어서 강의 종료가 되었을 때에도 학생 특성에 따라서 영향을 받지 않았음을 알 수 있다.

4.3 컴퓨팅 사고력 요소별 결과

컴퓨팅 사고력 요소에 대한 사전·사후 독립표본 검정을 실시한 결과는 <Table 5>와 같다.

Table 5. Computational Thinking Factor test result (p<.05)

Elements		N	Avg.	Std.	t	P
Pattern	Pre	116	1.79	.467	2.794	.006
	Post	116	1.59	.647		
Abstraction	Pre	116	2.19	.721	3.249	.001
	Post	116	1.90	.651		
Algorithm	Pre	116	1.87	.928	1.586	.114
	Post	116	1.66	1.055		
Automation	Pre	116	1.66	.591	.644	.520
	Post	116	1.60	.631		
Inference	Pre	116	1.40	.617	-.884	.377
	Post	116	1.47	.716		
Total	Pre	116	8.91	2.155	2.263	.025
	Post	116	8.22	2.421		

전체 컴퓨팅 사고력의 평균은 통계적으로 $p<0.05$ 수준에서 유의한 차이($t=2.263$, $p=0.025$)가 있는 것으로 나타났다. 이외에도 규칙($t=2.794$, $p=0.006$), 추상화($t=3.249$, $p=0.001$)도 유의미한 결과가 나타났다. 따라서 학생들에게 컴퓨팅 사고력 향상에 효과가 있음을 확인할 수 있었다. 다만 알고리즘($t=1.586$, $p=0.114$), 자동화($t=.644$, $p=0.520$), 추론($t=-.884$, $p=0.377$)은 유의미한 차이가 나타나지 않았는데, 이는 학생들이 수업 중에 반복문을 가장 어려워하고 있음을 증명하고 있다. 그러므로 향후 소프트웨어 교육 모델을 개선할 때에는 자동화, 알고리즘, 추론이 적용될 수 있는 다양한 반복 상황을 제시하여 반복문에 대한 개념을 익힐 수 있도록 준비할 필요가 있다.

5. 결론

다가오는 4차 산업혁명 시대를 대비하여 전공에 상관없이 누구나 갖춰야할 가장 중요한 능력으로 컴퓨팅 사고력이 강조되고 있다. 이를 위한 보편적인 교육으로서의 소프트웨어 교육은 컴퓨팅 기술을 이용한 문제해결의 경험을 흥미롭게 할 수 있도록 구성되어야 하지만, 아직도 많은 대학에서는 프로그래밍 언어 교육으로서의 한계를 넘어서지 못하고 있다.

본 논문에서는 컴퓨터 비전공자를 위한 고도의 분석과 접근에 기초한 소프트웨어 교육 모델을 제시하고, 타당성 검토를 실시하였다. 그 결과, 집단 특성에 따라서는 유의미한 차이가 나타나지 않았고, 컴퓨팅 사고력 평균

과 규칙성, 추상화 영역에서 유의미한 결과가 나타났다.

향후 연구로는 컴퓨터 비전공자를 위한 소프트웨어 교육을 실시하고 전공 영역별 특성에 따른 교육적 접근과 그에 따른 체계적인 관리가 필요하며, 소프트웨어 교육을 실시한 후에는 심층적인 분석을 통해 이후의 교육 전략에 반영하는 노력이 필요하다. 본 논문에서 제시한 교육 모델과 수업 전략들을 바탕으로 소프트웨어 교육이 이루어진다면 컴퓨터 비전공 학생들에게 보다 흥미로운 소프트웨어 교육이 될 것이며, 컴퓨팅 사고력 향상에 긍정적인 영향을 미칠 수 있을 것이다.

REFERENCES

[1] *Software Policy Division. (2015). Plan to educate human resources for software-oriented society, Ministry of Science and ICT. http://www.msit.go.kr/web/*msipContents/contentsView.do?cateId=mssw315&artId=1270998

[2] S. H. Jin & S. Shin. (2013) Case Study and Needs Analysis on Convergence Education in Engineering Colleges. *Journal of Engineering Education Research, 16,* 29-37.

[3] H. W. Lee, H. R. Min & K. W. Yi. (2008). A Study on the Improvable Proposal of General Education Curriculum of Engineering College - A Case of Seoul National University. *Journal of Engineering Education Research, 11(3),* 24-32.

[4] H. Y. Jung. (2014). An Empirical Study on Information Liberal Education in University based on IT Fluency and Computational Thinking Concept. *Journal of the Korea society of computer and information, 19(2),* 263-274.

[5] K. Kim & H. Kim. (2014). A Case Study on Necessity of Computer Programming for Interdisciplinary Education, *Journal of Digital Convergence, 12(11),* 339-348, 2014.

[6] Python Software Foundation. (2017). *Python about.* https://www.python.org/about/

[7] P. Guo. (2014). *Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities,* BLOG@CACM, http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext.

[8] C. W. Kim. (2010). The Application of Computer Education in the Revise Curriculum. *Journal of educational Research Institute, 12(1),* 41-55.

[9] S. Jun. (2017). Design and Effect of Development-Oriented Model for Developing Computing Thinking in SW Education. *Journal of The Korean Assocaition of Information Ecucation, 21(6),* 619-627.

[10] J. Kim et al,. (2016). 2015 Education Policy Network Training on-site support research : Development of SW Education Teaching and Learning Model. Commissioned research CR 2015-35.

[11] J. Seo. (2017). A Case Study on Programming Learning of Non-SW Majors for SW Convergence Education. *Journal of Digital Convergence, 15(7),* 123-132.

[12] S. H. Kim. (2015). Analysis of Non-Computer Majors' Difficulties in Computational Thinking Education. *The Journal of Korean Association of Computer Education, 18(3).* 15-23.

[13] Y. Sung. (2017). Development of SW Education Model based on HVC Learning Strategy for Improving Computational Thinking. *Journal of The Korean Assocaition of Information Ecucation, 21(5),* 583-593.

[14] S. H. Park. (2016). Study of SW Education in University to enhance Computational Thinking. *Journal of Digital Convergence, 14(4),* 1-10.

[15] Y. Lee & J. Cho. (2017). The Influence of Python Programming Education for Raising Computational Thinking. *International Journal of u- and e- Service, Science and Technology. 10(8),* 59-72.

[16] KERIS. (2016). *Research report KR 2016-4.* http://lib.keris.or.kr/search/detail/CATLAB000000012086.

이 영 석(Lee, Youngseok) [종신회원]

- 1999년 2월 : 서울교육대학교 초등교육과 (교육학사)
- 2001년 2월 : 서울교육대학교 컴퓨터교육과 (교육학석사)
- 2009년 8월 : 한양대학교 전자통신전파공학과 (공학박사)
- 2016년 3월 ~ 현재 : 강남대학교 KNU 참인재대학 컴퓨터 프로그래밍 교육 담당 교수
- 관심분야 : 소프트웨어 교육, 스마트러닝, 컴퓨터 프로그래밍, 지능형 웹 정보 시스템
- E-Mail : yslee38@kangnam.ac.kr