Journal of The Korean Association
of Information Education Vol. 22, No. 1,
February 2018, pp. 41-52

http://dx.doi.org/10.14352/jkaie.2018.22.1.41

비전공자 예비교사의 컴퓨팅 사고력 함양을 위한 프로그래밍 교육의 효과성 분석

한영신
인하대학교

요 약

컴퓨팅 사고력은 디지털 사회에서 핵심 인재로 성장하기 위한 기초 학습 능력으로서 강조되고 있다. 국가적으로 컴퓨팅 교육에 대해 많은 관심과 논의가 이루어지고 있으며, 이에 우리 정부도 컴퓨팅 교육을 교육과정에 포함시켰다. 이 같은 변화에 따라 예비교사들이 갖추어야 하는 컴퓨터 활용 능력의 수준이 높아지고, 디지털 사회에 맞추어 컴퓨팅 사고력 지도자로서의 역할이 교사들에게 강조되고 있다. 따라서 예비 교사 양성 교육과정에서부터의 다양한 프로그래밍 학습을 통해 컴퓨팅 사고력을 이해할 수 있는 교육과정이 필요하다. 본 연구에서는 비전공자 예비교사를 대상으로 한 프로그래밍 교육과정을 설계하고 교육을 진행하였으며, 개발한 설문 도구를 통해 비전공자 예비교사의 컴퓨팅 사고력 함양에 대한 프로그래밍 교육의 효과성을 분석하였다.

키워드 : 소프트웨어 교육, 컴퓨팅 사고력, 프로그래밍, 예비교사, 스크래치, 파이썬

Analysis of Effectiveness of Programming Learning for Non-science Major Preliminary Teachers' Development of Computational Thinking

Youngshin Han
Inha University

ABSTRACT

In Computational thinking is emphasized as a basic learning ability to grow into a key talent in the digital society. There is much interest and discussion on computing education nationally, and the Korean government has also included education in the curriculum. As a result of these changes, the level of computer literacy that preliminary teachers need to be imporved, and the role of computational thinking as a leader in digital society is being emphasized by teachers. Therefore, it is necessary to have a curriculum that can understand computational thinking through various programming learning from preliminary teacher education curriculum. In this study, we designed and taught programming curriculum for non-majored preliminary teachers. Through the developed questionnarie, we analyzed the affect of programming education on the preliminary teacher's development of computational thinking ability.

Keywords : Software Education, Computational Thinking, Programming, Preliminary Teacher, Scratch, Python

---

이 논문은 인하대학교의 지원에 의하여 연구되었음
교신저자 : 한영신(인하대학교)
논문투고 : 2017-12-05
논문심사 : 2017-12-07
심사완료 : 2017-12-27

1. 서론

컴퓨팅 사고력(Computational Thinking, CT)이란 컴퓨팅의 기본적인 개념과 원리를 기반으로 문제를 효율적으로 해결하는 사고 능력이다[16].

컴퓨팅 사고력은 컴퓨터 분야 뿐 만 아니라, 다양한 분야에 접목되어 발전하고 있다. 컴퓨팅 사고력은 컴퓨터 과학의 핵심이며 21세기 디지털 사회에서 핵심 인재로 성장하기 위한 기초 학습 능력으로서 강조되고 있다[15].

최근 창의적 문제 해결능력 및 논리적 사고력을 신장시키기 위하여 소프트웨어 교육의 필요성이 대두 되고 있으며, 특히 컴퓨팅적인 문제 해결능력 향상을 위한 프로그래밍 교육이 시행되고 있다[6].

전 세계적으로 컴퓨팅 교육에 대한 많은 관심과 논의가 있어왔다. 미국이나 영국, 이스라엘, 유럽 등 해외 여러 나라는 학생들의 컴퓨팅 사고력을 기르기 위해 초등학교 저학년부터 소프트웨어 교육을 실시하고 있으며[4], 이에 우리 정부도 컴퓨팅 사고력을 교육과정에 반영하기 위해 2015 개정 교육과정에 소프트웨어 교육을 포함시켰다[13].

이 같은 변화에 따라 예비교사들이 갖추어야 하는 컴퓨터 활용능력의 수준이 높아지고, 디지털 사회에 맞추어 새로운 지식을 재구성하고 주도적으로 학습할 수 있는 학습 촉진자로서의 역할이 교사들에게 강조되고 있다[10].

하지만, 현재 교사들은 소프트웨어 및 프로그래밍 교육을 받아본 경험이 전혀 없는 경우가 많으며, 학생들의 컴퓨팅 사고력에 대한 이해가 부족한 실정이다. 이 같은 상황을 해결하기 위해서는 예비교사 교육과정에서부터 변화가 필요하다. 현재 우리나라 예비교사 양성 교육과정에서는 ICT 소양 위주의 컴퓨터 교양 교육이 이루어지고 있으며, 컴퓨팅 사고력이나 소프트웨어와 관련한 교육은 부족하다[18].

미래의 인재들에게 컴퓨터 사고력과 소프트웨어 교육을 진행할 예비교사의 지도역량은 무엇보다 중요하다[10].

따라서 ICT 소양 위주의 컴퓨터 교양 교육 뿐 만 아니라 다양한 프로그래밍 학습을 통해 소프트웨어 교육과 컴퓨팅 사고력을 이해할 수 있는 교육과정이 필요하다.

본 연구는 비전공자 예비교사의 컴퓨팅 사고력 함양을 위한 프로그래밍 교육의 효과성을 분석하기 위한 연구이다. 본 연구에서는 비전공자 예비교사의 컴퓨팅 사고력 함양을 위해 프로그래밍 교육과정 설계 및 설문도구를 개발하였다. 설계된 프로그래밍 교육과정에서 사용되는 프로그래밍 언어는 스크래치(scratch)와 파이썬(python)이며, 비전공자가 프로그래밍 교육과정에서 겪을 수 있는 어려움을 고려하여 설계되었다. 설계된 교육과정은 사범대학 국어교육과, 사회교육과, 영어교육과 강의에 적용되었으며, 컴퓨팅 사고력과 프로그래밍 학습에 대한 인식 및 효과성을 측정하기 위한 설문조사는 프로그래밍 교육 전-후에 진행되었다.

본 논문의 구성은 1장의 서론에 이어 2장에서는 본 연구의 이해를 도울 수 있는 컴퓨팅 사고력 및 프로그래밍 교육과 관련한 연구 및 이론을 설명한다. 3장에서는 프로그래밍 교육과정을 설명하고, 4장에서는 연구방법 및 절차를 기술하기 위해 연구대상, 연구절차 등의 대해 기술한다. 5장에서는 사전검사와 사후검사 결과를 기술하여 연구 결과를 제시하고, 마지막으로 6장에서는 결론을 맺는다.

2. 관련연구 및 배경이론

2.1 선행연구 분석

본 연구와 관련된 선행 연구들에서는 프로그래밍 교육을 통해 학생들이 추상적인 개념을 이해하거나, 자신이 해결하고자 하는 문제에 대한 해결방법을 컴퓨팅 사고를 이용해 고안할 수 있다고 주장한다[8][11][17].

프로그래밍 교육은 컴퓨팅 사고력, 창의적 사고 및 논리적 사고 능력을 향상시키는데 도움이 될 수 있으나, 프로그래밍 언어가 일반 학습상황에서 교육적 효과를 가지기 위해서는 프로그래밍 언어를 배우는 과정에서의 어려움을 극복하는 것이 우선되어야 한다고 강조되고 있다[19].

2.2 국외 비전공자 대상의 프로그래밍 교육

국내외적으로 소프트웨어 교육에 대한 관심이 높아지고 있으며, 창의적인 융합 인재 양성을 위해 소프트웨어 비전공자들도 컴퓨터 과학에 대한 개념적 이해를 토대로 소프트웨어 설계 및 개발 할 수 있는 능력을 갖출 수 있어야 한다는 인식이 커지고 있다[12].

국외 대학은 비전공자들이 속하는 학문 분야에서의 소프트웨어 교육을 실시하고 있으며, 주로 컴퓨팅 기본 원리 및 컴퓨팅 사고와 프로그래밍 등이 공통적으로 구성된 교육과정이 운영되고 있다[19].

<Table 1>은 미국 워싱턴 대학교 비전공자를 위한 프로그래밍 입문 교육과정을 나타낸다[20].

<Table 1> The non-major course of Programming education at Washington University

Course	Overview
Computer Science Principle	Introduces fundamental concepts of computer science and computational thinking.
Computer Programming Ⅰ	Basic programming in the small abilities and concepts including procedural programming, basic control structures, etc.
Computer Programming Ⅱ	Concepts of data abstraction and encapsulation including stacks, queues, linked lists, binary trees, recursion,.
Web Programming	Covers languages, tools, and techniques for developing interactive and dynamic web pages.
Data Programming	Introduction to computer programming. Concepts of computational thinking, Python programming etc.
Currnet Topics in CSE	New or experimental courses that cover topics in computer science of contemporary relevance.

2.3 컴퓨팅 사고력

컴퓨팅 사고력이라는 개념은 Carnegie Mellon University의 Jannette W.Wing 교수에 의해 처음 소개 되었다. W.Wing은 컴퓨팅 사고력을 문제를 수립하고 해결책을 만들어 컴퓨팅 시스템을 통해 효과적으로 수행되도록 표현하게 하는 사고 과정이라고 정의하였다. 컴퓨팅 사고력은 컴퓨터의 해결 능력인 데이터 수집·분석, 표현, 문제 분해·추상화, 자동화 등을 사고에 적용시켜 여러 분야에서 문제를 해결하는데 필요한 능력으로 정의할 수 있다[16].

미국의 CSTA(Computer Science Teacher Association)와 ISTE(International Society for Technology in Education)는 컴퓨팅 사고력의 세부 구성 요소와 핵심 개념을 <Table 2>와 같이 9가지로 구분하였다[21].

<Table 2> Operational definition of key concepts of Computational Thinking

Component	Concepts
Data Collection	Gather resources to solve problems based on understanding and analyzing the problem
Data Analysis	Classify and analyze the collected data and the data given to the problem in detail
Data Representation	Represent the content of the problem in graph, chart, word, image, etc.
Data Decomposition	Analyze issues to solve problems
Abstraction	Define the definition of basic concepts to reduce the complexity of the problem
Algorithm & Procedures	Describe the steps to solve the problem in sequence
Automation	Select the best way to solve the problem by using the computing device in order to order and describe the contents of the problem
Simulation	Simulation to choose a complicated, difficult solution or viable solution
parallelization	Configuring resources to simultaneously perform tasks to achieve objectives

2.4 비전공자의 프로그래밍 교육

최근 문제 해결능력 및 논리적 사고력을 신장시키기 위하여 소프트웨어 교육의 필요성이 대두되고 있으며, 특히 컴퓨팅 사고력 향상을 위한 알고리즘과 프로그래밍 교육이 시행되고 있다[6].

하지만, 프로그래밍을 처음 접근하는 비전공자들은 프로그래밍 기술을 습득하는 과정에서 프로그래밍 언어들의 기본 문법, 로직 등을 파악하고 이해하는데 어려움을 겪고 있다[14].

Jenkins는 프로그래밍 교육과정에서 겪을 수 있는 어려움을 다음 아래와 같이 7가지로 분류하였다[5].

- Multiple Skills – programming is not a single skill. It is also not a simple set of skills.
- Multiple Processes – Programming is not only more than a single skill. it also involves more than one distinct process.
- The Language – Languages designed for serious use by serious programmers are hardly suitable

for the novice.

- Education Novelty - Programming is a new subject for many of the students who take programming courses.
- Interest - Learning programming can be very dull.
- Reputation and Image - Programming courses acquire the reputation of being difficult.
- Pace - Programming is taught, and therefore learned, to a set timescale.

프로그래밍에 대한 부정적인 태도와 자신감 상실 등은 다양한 전공의 예비교사들이 현장에서 교육을 시도하는데 장애 요인이 될 수 있다[18].

따라서 본 연구에서는 비전공자 예비교사가 프로그래밍에 대한 긍적적인 태도를 형성하기 위해 비전공자가 프로그래밍 교육과정에서 겪을 수 있는 어려움을 고려하여 프로그래밍 교육과정을 개발하였다.

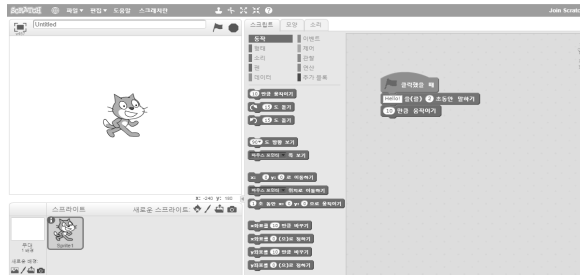
2.5 교육용 프로그래밍 언어

교육용 프로그래밍 언어는 컴퓨터 과학뿐만 아니라 수학이나 과학과 같은 다양한 교과에서 활용되고 있는 교수학습 도구이다[2]. 대표적인 교육용 프로그래밍 언어의 종류에는 엔트리(Entry), 앨리스(Alice), 로고(Logo), 스크래치(Scratch)등이 있다.

교육용 프로그래밍 언어가 C, Java와 같은 전문 프로그래밍 언어와 구분되는 큰 차이점은 단순한 프로그래밍 도구의 차원을 넘어 학습 환경을 제공한다는 점이다. 또한 학습자가 다루기 쉬운 형태로 구성되어 있어 학습자들의 인지적 부담을 감소시켜 주고, 풍부한 멀티미디어 콘텐츠 제작을 위한 저작 기능을 포함하고 있기 때문에 수학이나 과학과 같은 다양한 교과 교육을 위한 교수 학습 도구로 사용 될 수 있다[1].

본 연구에서는 스크래치를 이용한 프로그래밍 교육을 통해 비전공자 예비교사의 인지적 부담을 줄이고, 컴퓨팅 사고력을 향상시키고자 한다.

스크래치는 MIT 미디어 연구소의 Lifelong Kindergarten이 개발할 교육용 프로그래밍 도구이다. 스크래치는 명령들이 서로 반응하는 방식을 쉽게 제어할 수 있고 구문 오류로 인한 부정적인 피드백으로 인해 프로그래밍에 대한 두려움을 갖이 않도록 하는데 효과가 있다[9].



(Fig. 1) Scratch's development environment

스크래치의 개발 환경은 (Fig. 1)과 같다.

3. 프로그래밍 교육과정

본 연구에서는 비전공자 예비교사를 위한 프로그래밍 교육과정에 따라 한 학기 동안 교육을 지도하였다.

전체 교육과정은 15주로 구성되며 중간고사 및 기말고사를 제외하고 총 13주 동안 교육이 진행되었다. <Table 3>는 15주간의 전체 교육과정을 보여준다.

프로그래밍 교육과정에서 사용되는 프로그래밍 언어는 스크래치와 파이썬이다.

스크래치는 교육용 프로그래밍의 대표적인 언어로 학습자들이 쉽게 사용법을 익히고 응용할 수 있으며, 더나아가 프로그래밍 교육의 효과성을 높이고자 파이썬을 사용하였다. 파이썬은 전 세계적으로 비전공자들에게 교육되고 있는 일반 프로그래밍 언어 중 하나로 간결한 문법으로 입문자가 이해하기 쉽고, 다양한 분야에서 활용 될 수 있다는 장점을 가지고 있다.

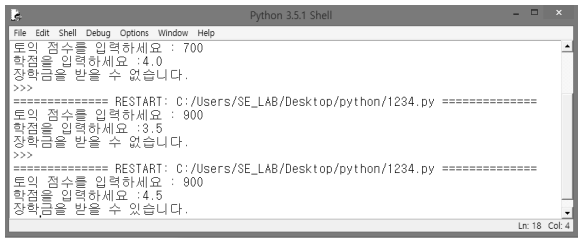
프로그래밍 교육과정은 비전공자가 프로그래밍 교육과정에서 겪을 수 있는 어려움을 고려하여 설계 되었다. 비전공자들은 주로 프로그래밍 언어의 문법 학습 , 프로그래밍에 대한 생소함, 전공과 무관함 등을 어려움으로 언급하였다. 본 연구에서는 이러한 점을 고려하여 프로그래밍의 기본 문법을 기초부터 학습할 수 있는 프로그래밍 교육과정을 설계하였다. 또한 프로그래밍에 대한 관심과 흥미를 높이기 위해 전공과 관련된 문제 해결과 풀이를 위주로 교육을 지도하였다.

(Fig. 2), (Fig. 3)은 프로그래밍 교육과정에서 스크래치와 파이썬을 각각 활용한 예시를 보여준다.

<Table 3> Programming Education Curriculum

Week	Learning	Contents	Overview
1	scratch	Learn scratch features and how to run them	- Introduction of Computational Thkinking - Learn about Scract - Scratch project Editor
2		Learn how to use scratch blocks	- Attributes of the scratch block - Combine scratch blocks
3		Learn about conditional blocks	- Getting to know the basic concept of conditional - Practice combining the conditional blocks - Solve and comment conditional statement examples realted to majors
4		Learn about loop blocks	- Getting to know the basic concept of loop - Practice combining the loop blocks - Solve and comment loop statement statement examples realted to majors
5		Learn about variable	- Getting to know the basic concept of variables - Creating conditional statements using variables - Creating loop statements using variables
6		Learn about function	- Getting to know the basic concept of function - Practice defining and calling functions - Define and call the functions related to majors
7		Middle test	- Middle test
8	python	Learn python features and how to run them	- Introduction of Programming Languages - Learn about python
9		Learn about variable & operator	- Getting to know the concept of variables - Declaring and defining variables - Getting to know the concept of operators(arithmetic, logical) - Using variables and operators
10		Learn about data type	- Learn about python's data type(integer, float, string)
11		Learn about list&sort	- Getting to know the concept of list & sort
12		Learn about conditional	- Getting to know the concept of conditional - IF, IF-else, IF-elif-else Statment - Solve and comment conditional statement examples realted to majors
13		Learn about loop	- Getting to know the concept of loop - Range, While, For Statment - Solve and comment loop statement examples realted to majors
14		Learn about function	- Getting to know the concept of function - Practice defining and calling functions - Define and call the functions related to majors
15		Final test	- Final test

(Fig. 2) Learn how to use scratch blocks



(Fig. 3) Learn about python's conditional

4. 연구방법 및 절차

4.1 연구대상

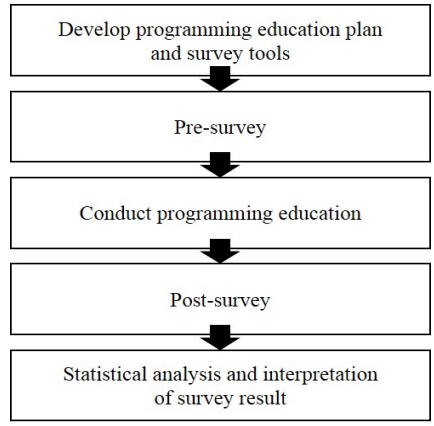
본 연구의 대상은 인천광역시 소재 I대학의 국어교육과, 사회교육과, 영어교육과 1학년에 재학중인 비전공자 예비교사 67명(국어교육과 11명, 사회교육과 27명, 영어교육과 29명)을 대상으로 연구를 실시하였다.

4.2 연구절차

본 연구의 절차는 (Fig. 4)와 같다.

1) 비전공자가 프로그래밍 교육에서 겪을 수 있는 어려움을 고려하여 프로그래밍 교육과정을 설계, 컴퓨팅 사고력과 프로그래밍 학습에 대한 인식 및 효과성을 측정하기 위해 설문 도구를 개발한다.

2) 프로그래밍 교육 전-후의 인식 및 효과성에 대한 차이를 알아보기 위해 프로그래밍 교육이 시작 되는 1주차에 사전 설문 조사를 실시한다.



(Fig. 4) Research Process

3) 프로그래밍 교육과정에 따른 교육을 지도한다.

4) 프로그래밍 교육 전-후의 인식 및 효과성에 대한 차이를 알아보기 위해 프로그래밍 교육이 끝나는 15주차에 사후 설문 조사를 실시한다.

5) 설문 결과를 바탕으로 한 통계적 분석을 통해 비전공자 예비교사의 컴퓨팅 사고력 함양을 위한 프로그래밍 교육의 효과성 정도를 분석하고, 결론을 도출한다.

4.3 연구도구

본 연구에서는 사범대학 학생들을 대상으로 하여 프로그래밍 학습 전-후의 컴퓨팅 사고력과 프로그래밍 학습에 대한 인식 및 효과성을 측정하기 위해 설문 도구를 개발하여 검사를 실시하였다.

개발된 설문은 총 20문항으로 컴퓨팅 사고력 영역과, 프로그래밍 영역으로 나뉜다. 컴퓨팅 사고력 영역은 컴퓨팅 사고력을 통한 문제 해결 능력 및 인지의 정도를 묻는 영역으로 (1) 컴퓨팅 사고력에 대한 해결능력 3문항, (2) 컴퓨팅 사고력에 대한 인지수준 7문항으로 구성되어 있다. 프로그래밍 영역은 프로그래밍을 통한 문제 해결능력 및 인지의 정도를 묻는 영역으로 (3) 프로그래밍에 대한 해결능력 3문항, (4) 프로그래밍에 대한 인지수준 7문항으로 구성되어 있다.

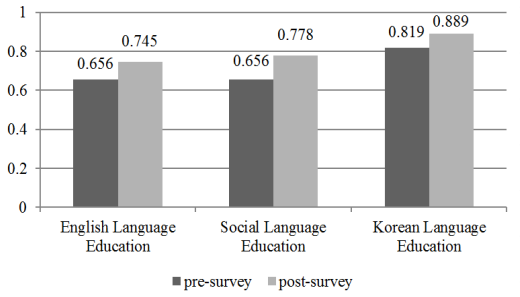
개발된 설문 도구의 컴퓨팅 사고력 영역에 대한 문항은 Jang[18]이 개발한 학습자의 컴퓨터 활용 능력검사 도구를 본 연구의 맥락에 맞게 수정하여 사용하였으며, 프로그래밍 영역의 문항은 비전공자를 대상으로 프로그

래밍 교육을 실시했다는 점을 고려하여 컴퓨터 공학 전문가와의 적절한 토의를 통해 개발하였다.

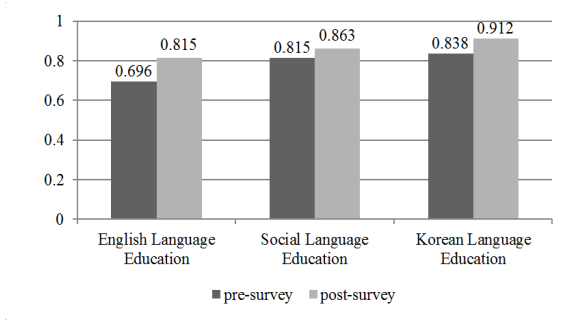
개발된 설문 도구의 문항은 IBM SPSS Statics 프로그램의 신뢰도 분석을 통해 결과를 분석하였다. <Table 4>는 영어교육과, 사회교육과, 국어교육과 학생들 대상으로 실시한 설문 항목에 대한 영역별 신뢰도 결과이다.

<Table 4> Confidence in the survey

Department	Domain	Cronbach's α (pre)	Cronbach's α (post)
English Language Education	Computational Thinking	0.656	0.745
	Programming	0.696	0.815
Social Studies Education	Computational Thinking	0.656	0.778
	Programming	0.815	0.863
Korean Language Education	Computational Thinking	0.819	0.889
	Programming	0.838	0.912



(Fig. 5) Confidence of Computational Thinking



(Fig. 6) Confidence of Programming

4.4 자료 처리 방법

각 문항의 답변은 Likerttechniqu(리커트법) 5점 척도로 구성되어있으며, '매우 그렇다'는 5점, '그렇다'는 4점, '보통이다'는 3점, '그렇지 않다'는 2점, '전혀 그렇지 않다'는 1점으로 하여 분석하였다.

설문조사의 결과가 사전-사후에 차이가 있는지 대응표본 t검정을 통해 통계적으로 검정하였으며, 통계 분석은 IBM SPSS Statistics 프로그램을 사용하였다.

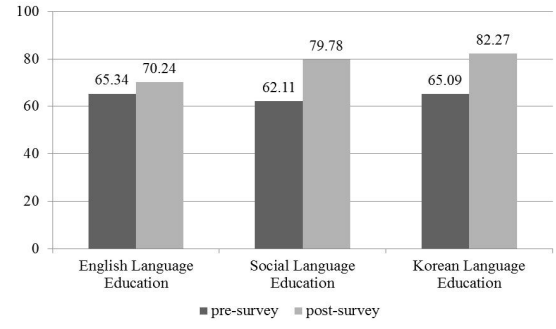
5. 연구결과

프로그래밍 학습 전 후의 컴퓨팅 사고력과 프로그래밍 학습에 대한 인식의 변화를 살펴보기 위해 학기 초와 학기 말에 검사를 시행하였다.

사전 검사와 사후 검사는 동일한 설문지로 시행하였으며, 설문의 결과 값에 대하여 대응표본 t검정을 수행하였다. 대응표본 t검정의 분석 내용은 <Table 5>와 같다.

설문검사의 전체의 평균은 영어교육과 사전검사 65.34, 사후검사 70.24, 사회교육과 사전검사 62.11, 사후검사 79.78, 국어교육과 사전검사 65.09, 사후검사 82.27로 나타났다. 영어교육과, 사회교육과, 국어교육과에서 모두 사후검사의 평균점수가 상승하였고, 유의수준에서도 역시 영어교육과 0.021, 사회교육과 0.000, 국어교육과 0.006으로 통계적으로 유의미한 값을 나타내고 있다.

컴퓨팅 사고력과 프로그래밍 영역에서 모두 평균은 영어교육과, 사회교육과, 국어교육과 모두 사후검사의 평균



(Fig. 7) Paired t-test result by department

<Table 5> Paired t-test result

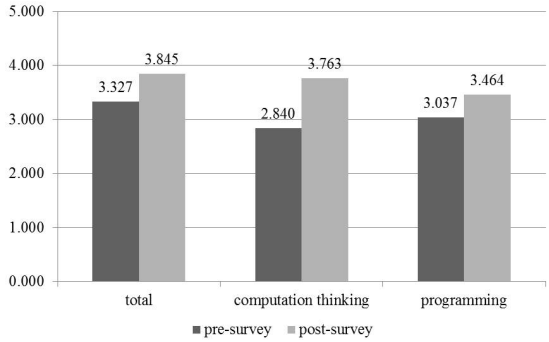
Department	Domain	Survey	*N*	*M*	*SD*	*t*	*p*
English Language Education	All	Pre	29	65.34	8.376	-2.44	0.021
		Post	29	70.21	8.042		
	Computational Thinking	Pre	29	33.86	4.665	-2.069	0.048
		Post	29	36.10	3.609		
	Programming	Pre	29	31.48	4.076	-2.461	0.020
		Post	29	34.10	5.017		
Social Studies Education	All	Pre	27	62.11	11.603	-6.576	0.000
		Post	27	79.78	8.252		
	Computational Thinking	Pre	27	31.89	6.399	-5.072	0.000
		Post	27	40.07	3.980		
	Programming	Pre	27	30.22	5.767	-7.484	0.000
		Post	27	39.70	5.067		
Korean Language Education	All	Pre	11	65.09	12.637	-3.45	0.006
		Post	11	82.27	11.163		
	Computational Thinking	Pre	11	33.73	7.250	-3.517	0.006
		Post	11	41.55	4.610		
	Programming	Pre	11	31.73	8.956	-2.478	0.033
		Post	11	39.82	5.250		

<Table 6> Average and pre-post comparison of the questions of computational thinking

		Question	*Survey*	*N*	*M*	*SD*	*t*	*p*
Ability to solve	1	I can logically design the process to solve the problem.	Pre	67	3.12	0.930	-4.426	0.000
			Post	67	3.81	0.723		
	2	I can structure the problem in a form that can be solved through a program.	Pre	67	2.58	1.017	-6.882	0.000
			Post	67	3.70	0.739		
	3	I can express it with an algorithm that solves problems of everyday life.	Pre	67	2.82	1.167	-5.314	0.000
			Post	67	3.78	0.755		
Cognitive level	4	I feel the difficulty of doing computational thinking.	Pre	67	3.49	0.877	3.559	0.001
			Post	67	2.91	0.996		
	5	I think that computational thinking is necessary.	Pre	67	3.76	0.889	-2.344	0.022
			Post	67	4.09	0.596		
	6	I think (elementary, middle, high) school students need computational thinking skills.	Pre	67	3.73	0.931	-2.705	0.009
			Post	67	4.13	0.672		
	7	I think that computing thinking has improved through programming.	Pre	67	3.64	0.916	-3.062	0.003
			Post	67	4.09	0.690		
	8	I think that computational thinking helps to solve problems.	Pre	67	3.39	0.953	-4.354	0.000
			Post	67	4.01	0.769		
	9	I expect that computational thinking can develop the ability to plan effective class.	Pre	67	3.37	0.967	-3.686	0.000
			Post	67	3.96	0.747		
	10	I expect that computational thinking helps to accomplish my curriculum.	Pre	67	3.37	0.951	-4.092	0.000
			Post	67	3.97	0.778		

점수가 상승하였고, 유의수준에서도 역시 통계적으로 유의미한 값을 나타내고 있다. 따라서 프로그래밍 교육을 통해 컴퓨터 사고력이 향상 된 것으로 해석할 수 있다.

5.1 컴퓨팅 사고력에 대한 분석 결과



(Fig. 8) Average comparison of the questions of computational thinking

<Table 6>는 컴퓨팅 사고력에 대한 대응 표본 t검정 결과 값을 보여준다.

컴퓨팅 사고력에 대한 문항은 해결능력과 인지수준 영역으로 나누어 컴퓨팅 사고력에 대한 해결능력 3문항, 컴퓨팅 사고력에 대한 인지수준 7문항으로 구성된다.

컴퓨팅 사고력에 대한 문항 전체의 사전검사 평균은 3.327, 사후검사 평균은 3.845이다. 유의수준에서도 역시 모든 문항에서 0.05에서 통계적으로 유의미한 값을 나타낸다.

컴퓨팅 사고력의 해결능력 영역에 대한 문항 전체의 사전검사 평균은 2.840, 사후검사 평균은 3.763이며, 인지수준 영역에 대한 문항 전체의 사전검사 평균은 3.037, 사후검사 평균은 3.464이다.

컴퓨팅 사고력에 대한 총 10개의 문항에서 4번 문항을 제외한 9개 문항 모두에서 사후검사 평균점수가 사전검사 평균 점수 보다 상승했다.

문항별로 살펴보면, 4번 문항 '나는 컴퓨팅 사고를 하는 것에 어려움을 느낀다.'의 사후검사 평균 점수는 2.91으로 사전검사 평균 점수 3.49보다 0.58 감소하였다. 프로그래밍 교육을 통해 컴퓨팅 사고를 하는 것에 대해 어려움을 느끼는 정도가 감소하였음을 확인할 수 있다. 2번 문항 '나는 해결하고자 하는 문제를 프로그램을 통해 해결할 수 있는 형태로 구조화 시킬 수 있다.'의 사후검사 평균점수는 3.7로 사전검사 평균 점수는 2.58보다 1.12 상승하였다. 해결하고자 하는 문제를 프로그램을 통해 효율적으로 해결할 수 있는 능력이 향상되었다고 해석할 수 있다.

5.2 프로그래밍에 대한 분석 결과

<Table 7>는 프로그래밍에 대한 대응 표본 t검정 결과 값을 보여준다. 프로그래밍에 대한 문항은 해결능력과 인지수준 영역으로 나뉘어 프로그래밍에 대한 해결능력 3문항, 프로그래밍에 대한 인지수준 7문항으로 구성된다.

프로그래밍에 대한 문항 전체의 사전검사 평균은 3.100, 사후검사 평균은 3.730이다. 유의수준에서도 역시 모든 문항에서 0.05에서 통계적으로 유의미한 값을 나타낸다.

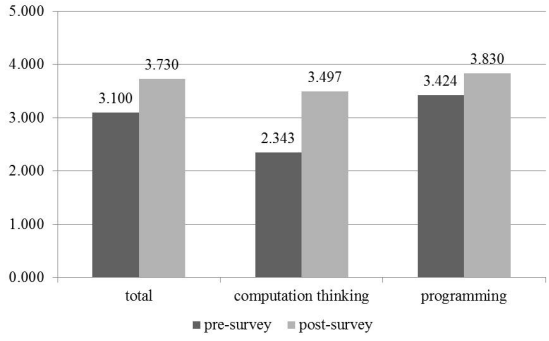
프로그래밍 해결능력 영역에 대한 문항 전체의 사전검사 평균은 2.343, 사후검사 평균은 3.497이며, 인지수준 영역에 대한 문항 전체의 사전검사 평균은 3.424, 사후검사 평균은 3.830이다.

프로그래밍에 대한 총 10개의 문항에서 4번 문항을 제외한 9개 문항 모두에서 사후검사 평균 점수가 사전검사 평균점수 보다 상승했다.

문항별로 살펴보면, 4번 문항 '나는 프로그래밍 언어를 배우는데 어려움을 느낀다.'의 사후검사 평균 점수는 2.90으로 사전검사 평균 점수 3.81보다 0.91 감소하였다. 프로그래밍 교육을 통해 프로그래밍 언어를 배우는 것에 대해 어려움을 느끼는 정도가 감소하였음을 확인할 수 있다. 2번 문항 '나는 다양한 프로그래밍 언어를 사용할 수 있다.'과 3번 문항 '나는 다양한 프로그래밍언어를 이용하여 해결하고자 하는 문제를 해결할 수 있다.'은 사후검사 평균 점수가 사전검사 평균 점수보다 1.00이상 상승하였다. 프로그래밍 교육을 통해 다양한 프로그래밍 언어를 사용할 수 있게 되었으며, 이를 이용하여 해문제를 해결할 수 있는 능력이 향상 되었다고 해석할 수 있다.

<Table 7> Average and pre-post comparison of the questions of programming

		Question	Survey	N	M	SD	t	p
Ability to solve	1	I can use various computer programs.	Pre	67	2.54	1.020	-6.203	0.000
			Post	67	3.49	0.842		
	2	I can use various programming languages.	Pre	67	2.22	1.056	-7.560	0.000
			Post	67	3.43	0.857		
	3	I can solve the problem using various programming languages.	Pre	67	2.27	1.009	-7.631	0.000
			Post	67	3.57	0.891		
Cognitive level	4	I have difficulty learning a programming language.	Pre	67	3.81	0.783	5.787	0.000
			Post	67	2.90	1.046		
	5	I am interested in learning programming.	Pre	67	2.90	0.923	-6.566	0.000
			Post	67	3.79	0.769		
	6	I think that programming learning is necessary.	Pre	67	3.45	0.840	-3.164	0.000
			Post	67	3.90	0.819		
	7	I think preliminary teachers need to learn programming.	Pre	67	3.23	0.943	-4.634	0.000
			Post	67	4.00	0.953		
	8	I think (elementary, middle, high) school students need programming skills.	Pre	67	3.54	0.959	-4.029	0.000
			Post	67	4.12	0.749		
	9	I think that programming learning will help improve logical thinking skills.	Pre	67	3.49	0.823	-3.852	0.000
			Post	67	4.00	0.739		
	10	I think that programming learning will help improve computational thinking skills.	Pre	67	3.55	0.875	-3.902	0.000
			Post	67	4.10	0.781		



(Fig. 9) Average comparison of the questions of programming

6. 결론

컴퓨팅 사고력은 컴퓨팅의 기본적인 개념과 원리를 기반으로 여러 분야에서 문제를 해결하는데 필요한 능력이다.

최근 디지털 사회에서 핵심 인재로 성장하기 위한 기초 학습 능력으로서 컴퓨팅 사고력이 강조되고 있으며, 특히 컴퓨팅적인 문제 해결능력 향상을 위해 프로그래밍 교육이 시행되고 있다.

이러한 흐름에 맞추어 새로운 지식을 재구성하고 컴퓨터 활용 능력을 갖춘 학습 촉진자로서의 역할이 교사들에게도 강조되고 있다.

하지만, 우리나라 예비교사 양성 교육과정은 ICT 소양 위주의 컴퓨터 교양 교육으로 구성되어 있으며, 컴퓨팅 사고력 능력을 갖추기 위하여 관련된 교육 내용은 부족한 실정이다.

따라서 예비 교사 양성 교육과정에서부터의 다양한 프로그래밍 학습을 통해 컴퓨팅 사고력을 이해할 수 있는 교육과정이 필요하다.

본 연구에서는 비전공자 예비교사의 컴퓨팅 사고력에 대해 프로그래밍 교육의 효과성을 분석하기 위해 수행되었다. 프로그래밍 교육과정은 비전공자가 프로그래밍 교육에서 겪을 수 있는 어려움을 고려하여 설계되었으며, 프로그래밍 언어로는 스크래치와 파이썬을 사용하였다. 또한 프로그래밍 교육의 효과성을 분석하기 위해 설문도구를 개발하여 프로그래밍 교육 사전-사후에 인식 및 효과성을 측정하는 검사를 진행하였다.

본 연구 결과 학생들은

첫째, 다양한 컴퓨터 프로그램을 통해 다양한 프로그래밍 언어를 사용할 수 있는 능력이 향상되었다. 둘째, 프로그래밍 교육을 통해 컴퓨팅 사고를 하는 것에 대해 어렵지 않다는 인식을 가지게 되었으며, 해결하고자 하는 문제를 해결할 수 있는 형태로 구조화 시킬수 있는 능력이 향상되었다. 셋째, 프로그래밍 언어를 배우는 것에 대해 어렵지 않다는 인식을 가지게 되었으며, 다양한 프로그래밍 언어를 이용하여 문제를 해결 할 수 있는 능력이 향상되었다. 넷째, 프로그래밍에 대한 관심과 필요성 정도가 향상 되었으며, 프로그래밍 교육을 통해 컴퓨팅 사고력이 향상 되었다.

이 결과들은 프로그래밍 교육이 비전공자 예비교사에게 컴퓨팅 사고 능력 함양에 있어 도움이 된다는 것을 나타내며, 긍정적인 효과를 가져오는 것을 보아 예비 교사를 대상으로 한 새로운 소프트웨어 교육과정으로 활용되기에 적합하다고 이야기 할 수 있다.

컴퓨팅 교육에 대한 많은 관심과 논의가 이루어지고 있다. 미래의 인재들에게 컴퓨팅 교육을 지도할 예비교사의 지도역량은 무엇보다 중요하다. 하지만 예비교사 양성 교육과정에 있어 컴퓨팅 교육과 관련한 교육과정은 부족한 실정이다. 앞으로 비전공자 예비교사들의 다양한 프로그래밍 학습 및 컴퓨팅 사고력 이해를 위한 학습 경험이 제공되기 위한 교육과정 개선에 있어 많은 노력이 필요하다고 생각된다.

참고문헌

[1] GyeongMi Ahn (2010). The Effect of the Programming Education on the Elementary School Student's Learning-Flow and programming ability. *(Unpublished master dissertation),* Gyeongin National University of Education, Korea.

[2] HakJin Bae, EunKyoung Lee, & YoungJun Lee (2009). A Problem Based Teaching and Learning Model for Scratch Programming Education, *The Journal of the Korean Association of Computer Education, 12*(3), 11-22.

[3] HuiSeon Jang (2013). Impact analysis on Netlogo programming advantage of learning motivation and learning ability. *(Unpublished master dissertation),* Ajou University of Education, Korea.

[4] International Society for Technology in Education & Computer Science Teachers Association (2011). CSTA K-12 Computer Science Standards Revised 2011.

[5] Jenkins, Tony (2002). On the difficulty of learning to program. *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, 4*(2002).

[6] JeongWon Choi, & YoungJun Lee (2014). The Design of Method for Evaluating Computational Thinking. *In Proceedings of Korean society of computer and information, 22*(2), 177-178.

[7] JinGee Ku (2017). A Study on the Software Convergence Education for Non-Majors Computer Science using Creative Robot. *The Journal of the korea Academia-Industrial cooperation society, 18*(2), 631-638.

[8] JungHyun An (2010). Scratch programming instruction model considering the characteristics of middle school students. *(Unpublished master dissertation),* Korea National University of Education, Korea.

[9] KiBbm Lee & SeokJae Moon (2016). Study on Basic Learning of Programming based on WeDo with Scratch for the Non-Science Major. *The Journal of the Society of Convergence Knowldege, 4*(2), 9-15.

[10] KongJu Mun, JiYeong Mun, SeMi Kim, & SungWon Kim (2016). Application of Programming Curriculum for Pre-service Science Teacher and Examination of their Perceptions about Programming. *The Journal of Learner-Centered Curriculum and Instruction, 16*(10), 825-842.

[11] KyungKyu Kim & JongYun Lee (2016). Analysis of the Effectiveness of Computational Thinking-Based Programming Learning. *The Journal of the Korean Association of Computer Education, 19*(1), 27-39.

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

[13] Ministry of Education (2015). *Guide of operating SW education.*

[14] SooHwan Kim (2015). Analysis of Non-Computer Majors' Difficulties in Computational Thinking Education. *The Journal of the Korean Association of Computer Education, 18*(3), 44-57.

[15] TaeHun Kim (2015). STEAM education program based on programming to improve computational thinking ability. *(Unpublished doctoral dissertation),* Jeju National University, Korea.

[16] Wing, Jannette M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

[17] Yohan Hwang, Kongju Mun & Yunenbae Park (2016). Study of Perception on Programming and Computational and Attitude toward Science Learning of High School Students through Software Inquiry Activity : Focus on using Scratch and physical computing materials. *The Journal of the korean Association for Science Education, 36*(2), 325-335.

[18] YongJu Jeon, & TaeYoung Kim (2016). The design of Computational Thinking-based Web programming course as a liberal art for Non-Computer Majored Preliminary Teachers. *Proceedings of KSCI Conference 2015,* 161-164.

[19] Zion Hwang & Sungon Hwang (2017). An Analysis of Research Trends Software Education for Elementary school : Focusing on Domestic Articles. *The Journal of the Korean Assocaition of Information Ecucation, 21*(5), 509-525.

[20] Computer Science & Engineering University of Washington (Non-Major Course Options). Retrieved December 1, 2017, from http://www.cs.washington.edu/prospective_students/undergrad/admissions/nonmajor

[21] International Society for Technology in Education (ISTE) & Computer Science Teachers Association (CSTA) (2011). Computational thinking in K - 12 education teacher resource. Retrieved December 1, 2017, fromhttp://www.iste.org/docs/ct-documents/ct-teacher-resources_2ed-pdf.pdf?sfvrsn=2.

저자 소개

한 영 신

2004 성균관대학교 전기전자컴퓨터공학과 공학박사

2004 이화여자대학교 컴퓨터그래픽&가상현실연구센터 박사후 연구

2005 성결대학교 멀티미디어학과 전임강사

2007 아리조나대학교 ACIMS센터 Visition Scholar

2009 성균관대학교 정보통신공학부 반도체시스템공학과 연구교수

2013~2017 성결대학교 컴퓨터공학부 조교수

2017~현재 인하대학교 프런티어학부 교수

관심분야: 컴퓨팅 사고, 알고리즘, 프로그래밍

e-mail: hanys@inha.ac.kr