인문계열 학생을 위한 SW교육에서의 초보 학습자 특성 분석

성정숙† · 김수환†† · 김현철†††

요 약

새로운 디지털 시대가 도래하고 학문과 산업에서 컴퓨팅(computing) 기반의 융복합적 성격을 띠는 분야가 많아지면서 컴퓨팅 사고력(computational thinking)의 중요성이 강조되고 있다. 컴퓨팅 사고력 함양을 위해서는 프로그래밍 교육이 매우 중요한데, 지금까지의 프로그래밍 교육은 전문가 양성을 목적으로 한 경우가 많았기 때문에 보편적 교육으로서의 프로그래밍 교육에 대한 면밀한 연구가 필요한 시점이다. 본 연구는 비전공자를 대상으로 하는 프로그래밍 교양 수업에서 프로그래밍을 처음으로 접하는 학습자들이 프로그래밍 초보 학습단계에서 보이는 흥미도, 도구 용이성, 자신감, 숙련도 등의 변화에 대해 설문, 관찰 및 인터뷰 방식을 통해 얻어진 결과를 토대로 탐색적 연구를 실시함으로써 모든 학생들을 대상으로 하는 보편적 프로그래밍 교육을 설계하고 수행하는 데에 도움을 주고자 한다.

주제어 : 컴퓨팅 사고력, 프로그래밍 초보학습자, 보편적 프로그래밍 교육, SW교육

Analysis of Art and Humanity Major Learners' Features in Programming Class

Jung Sook Sung† · Soo Hwan Kim†† · Hyeoncheol Kim†††

ABSTRACT

In digital era, as various fields of knowledge and industry are fused by computing, fostering computational thinking and learning computer programming are strongly emphasized. It means it is important to study how to educate computer programming for all, This study is about analysis of non computer science major learners' behavior gathered from computer programming class by means of survey, observation and interview and hope to suggest the way how to design and to perform the new computer programming curriculum for all.

Keywords : computational thinking, programming novice, computer education for all

† 정 회 원: 고려대학교 컴퓨터교육학과 박사과정
†† 종신회원: 충신대학교 교양교직과 조교수
††† 종신회원: 고려대학교 정보대학 컴퓨터학과 교수(교신저자)
논문접수: 2015년 2월 1일, 심사완료: 2015년 3월 11일, 게재확정: 2015년 4월 19일

1. 서론

산업혁명에 의하여 성립하고 발전해온 '산업사회'가 고도화되어 정보혁명을 통해 '정보사회'라는 새로운 단계로 전환되자 사회의 운영이 소프트웨어를 중심으로 이루어지게 되고, 산업과 학문의 많은 분야들이 컴퓨팅(computing) 기반의 융복합적 성격을 띠게 되었다[1][2]. 이러한 변화로 인해 정보처리의 관점으로 문제를 이해하고 효과적이고 효율적으로 문제를 해결할 수 있는 능력인 컴퓨팅 사고력(computational thinking)의 함양이 컴퓨터관련 직업을 가진 사람뿐 아니라 일반인들에게도 미래를 살아가는 데 필요한 핵심 역량으로써 중요성이 강조되고 있다[3][4].

컴퓨팅 사고력의 중요성이 강조되면서 컴퓨팅 사고력에 대한 연구 및 이를 강화하기 위한 교육방법에 대한 연구도 활발히 진행되기 시작하였다[5][6][7]. 최근에는 수학이나 과학 분야에서도 컴퓨팅 사고력을 위한 다양한 교육방법들이 연구되고 있으나, 컴퓨팅 사고력의 개념과 정의가 컴퓨터 과학의 기본원리를 바탕으로 출발하였으므로 컴퓨팅 사고력의 함양을 위해서 프로그래밍 교육을 하는 것이 가장 효과적이라고 할 수 있다[7].

프로그래밍 교육의 중요성에 대한 인식이 커짐에 따라 세계 선진국들은 프로그래밍 교육을 국가정규교육과정에 편입시키거나 민간단체와 교사, 시민들의 자발적인 활동으로 프로그래밍 교육 운동의 열풍이 확산되고 있는 상황이다[8].

그러나 프로그래밍을 처음 접하는 학생들은 어려움을 호소하는 경우가 많으며, 중도에 포기하는 학생도 많이 나타난다[9][10]. 그렇기 때문에 모든 학생들을 대상으로 하는 프로그래밍 교육을 설계하기 위해서는 전공을 목적으로 하지 않는 프로그래밍 초보자가 겪는 심리적 상황과 행동에 대한 분석과 이해가 반드시 선행되어야 한다.

따라서 본 연구는 전산과학이나 컴퓨터공학을 전공으로 하지 않는 비전공자이면서 프로그래밍을 처음으로 접하는 학생들을 대상으로 프로그래밍 교육을 했을 때 나타나는 초보학습자의 심리적 변화와 행위의 특징을 살펴보고 설문과 인터뷰를 통해 나타나는 현상을 탐색해보고자 한다.

2. 이론적 배경

2.1 보편적 프로그래밍 교육

Douglas Rushkoff(2010)는 디지털 시대를 살아가기 위한 열 가지 조언을 담은 「Program Or Be Programmed」에서 모든 디지털 테크놀로지는 프로그램이 되어 있으므로 이를 이해하고 다룰 수 있지 않으면 자신도 모르는 사이에 통제되고 조정 당하게 된다고 경고한다. 그렇기 때문에 프로그래밍을 배우는 것은 컴퓨터 공학 전공자에게만 필요한 것이 아니라 디지털 시대를 살아가는 모든 사람들이 배워야 하는 기본 지식이라고 설명한다[11].

모든 학생을 대상으로 하는 보편 교육으로서의 프로그래밍 교육이 성공하기 위해서는 언어 자체를 배우는 데 너무 많은 시간을 소요하거나 기술 전수에 집중되지 않아야 한다. 또한, 프로그래밍에 대해 특별한 능력이나 관심을 갖는 학생들이 아닌 일반 학생들이 겪을 수 있는 다양한 심리적 행위적 변화에 대해 연구가 이루어져야 한다.

Piteira(2013)는 프로그래밍을 배우는 데 있어서의 주요 어려움에 대해 학습내용, 학습상황, 학습자료의 세 부분으로 나누어 분석하였다. 그 결과 학습내용 면에서는 포인터와 참조, 파라미터, 배열이나 레코드와 같은 구조적 데이터 타입(structured data type), 추상적 데이터 타입(abstract data type), 예외 처리 등에서 어려움이 큰 것으로 나타났으며, 학습상황 면에서는 실습실에서 실습하는 시간을 가장 유용하게 느끼는 것으로 나타났고, 학습자료 면에서는 예제 코드를 가장 유용하게 느끼는 것으로 나타났다[12].

이러한 연구들은 프로그래밍 도구사용의 미숙함보다는 프로그래밍 개념에 대한 이해부족이 초보 학습자로 하여금 프로그래밍 학습을 어렵게 하는 원인으로 분석하고 있으며, 이를 위해서는 이론수업보다는 실제적인 예제 제시나 실습 시간 확보가 중요하다고 주장한다. 이렇게 학습자들이 느끼는 어려움의 원인 분석에 관한 연구는 종종 이루어지고 있으나 초보 학습자들이 프로그래밍 수업에 참여하는 동안 심리적, 인지적으로 어떠한 변화를 겪는지에 대한 연구는 부족한 상황이다.

2.2 스크래치

프로그래밍 교육을 위해서는 학습자와 학습목표에 맞는 적절한 프로그래밍 언어를 선택하는 것이 매우 중요하다. 이를 위해 현재까지 많은 종류의 다양한 교육용 프로그래밍 언어가 개발되어 있으나[13] 본 연구에서는 명령어가 시각적으로 표현되어 문법의 이해에 대한 부담이 적고 무료로 제공되며 단시간 내에 가르치기에 적당한 스크래치를 도구로 선택하였다.

스크래치는 미국의 MIT에서 초등학생을 위해 개발한 것으로 요철식 블록형 놀이방식을 이용한 프로그래밍 언어이다. 언어를 마우스의 드래그 앤 드롭방식을 이용하여 쉽게 작성할 수 있을 뿐 아니라 실행결과가 옆의 창에 시각적으로 바로 나타나기 때문에 텍스트 명령어를 기반으로 하는 언어들에 비해 어린이들이 상대적으로 쉽게 익힐 수 있는 장점이 있다.

또한 모든 소스코드가 공개되기 때문에 많은 예제들이 지속적으로 제공되고 공유될 수 있다는 점과 이 모든 것이 무료로 제공된다는 점에서 전 세계적으로 폭발적인 인기를 얻고 있다. 2007년 처음 개발된 이래 2014년 말 현재 450만명이 등록되어 있으며 700만개의 프로젝트가 공유되고 있다[14].

2.3 컴퓨팅 사고력 (Computational Thinking)

최근 세계적인 컴퓨터 교육의 추세는 특별한 집단을 대상으로 하는 수월성 교육이나 직업 교육에서 일반인의 소양을 위한 보편적 교육으로 변하고 있다. 또한 교육내용에 있어서도 기존에는 ICT(Information Communication Technology)를 활용하는 교육이 주를 이루었는데, 최근에는 컴퓨팅 사고력(Computational Thinking) 함양에 더 큰 비중을 두고 있다[15].

20세기말까지만 해도 모든 기관에서 21세기에 가장 필요한 역량 중 하나로 ICT 리터러시 혹은 디지털 리터러시를 선정하였다. 그러나 이제 ICT를 능숙히 사용하는 것만으로는 디지털 정보의 연결과 공유로 변화하고 있는 세상의 문제들을 해결하는 데 부족하다고 보고, 소프트웨어의 사용뿐 아니라 제작까지 할 수 있는 확대된 의미인 컴퓨팅 리터러시의 중요성을 강조하고 있다[15].

컴퓨팅 사고력이라는 용어는 2006년 Jeannette M. Wing[3]의 발표로 널리 알려진 이후로 여러 차례 정의와 개념에 대한 연구가 발표되었으며[16][17], 이를 종합하여 분석한 2014년 한국과학창의재단의 보고서에 의하면 '컴퓨팅 사고력은 컴퓨팅 시스템의 역량을 활용하여 해결하고자 하는 문제를 효과적이고 효율적으로 해결할 수 있는 절차적 사고능력'이라고 정의하고 있다[7].

따라서 컴퓨팅 사고력은 실생활에서 일어나는 단순한 문제에서부터 현실적으로 다루기 어려운 복잡한 문제에 이르기까지 해결하고자 하는 다양한 문제들을 작은 단위로 분해하고 절차적으로 나열하며 대량의 반복적인 행위를 자동화함으로써 문제를 해결해나갈 수 있는 능력을 의미한다.

이렇게 컴퓨터를 이용하여 문제를 해결하는 능력인 컴퓨팅 사고력에는 자동화할 수 있는 능력 포함되므로 이를 함양하기 위해서는 프로그래밍 교육이 매우 중요하다. 그러나 수업내용이 프로그래밍 도구 사용을 익히는 데에만 집중될 경우 학습의 목표가 도구 사용법을 익히는 데에만 머물고 사고력 함양으로 이어지지 않을 수 있으므로, 학습자가 컴퓨팅 사고력을 키우는 데 집중할 수 있도록 학습자에게 적절한 도구를 선택하고 수업과 실습 내용을 구성하는 것이 매우 중요하다.

3. 연구 방법

3.1 연구 대상

C대학교 멀티미디어개론 수업에 적용된 스크래치 실습수업은 2014년 4월부터 6월까지 총 7회 수행되었으며 전체 참여 학생은 71명이다.

실습수업 이전에는 컴퓨터 과학의 기초개념을 다루는 이론 수업을 진행하였다.

참여 학생들은 인문사회계열 전공자로 구성되어 있으며 이공계 전공자는 없었다. 전체 수강자 71명 중 본 수업을 듣기 전에 스크래치를 사용해 본 경험이 있는 학생은 5명이었다.

3.2 연구 방법

본 연구에서는 질적인 접근과 양적인 접근을 함께 활용해 자료를 수집 및 분석하고, 결과를 통합해 추론을 이끌어내는 혼합적 연구방법[18]을 사용하였다. 설문, 관찰일지, 인터뷰의 세 가지 연구방법을 사용하였으며 질적 연구의 일반화 가능성을 높이기 위해 관찰과 인터뷰에 두 명의 연구진이 참여하여 결과를 통합하여 이끌어내는 삼각측량법(triangulation)을 사용하였다.

설문은 대표적인 양적연구방법으로, 본 연구에서는 학습내용면에서 학생들이 느끼는 흥미도와 도구용이성, 자신감 등의 변화를 알아보기 위한 설문을 매 수업시간마다 조사하여 변화양상을 알아보았다.

관찰일지는 연구자가 현장에서 경험하게 되는 모든 사건을 기록한 것으로 단순한 내용만을 기록하는 것뿐 아니라 연구자의 경험, 기분, 느낌, 주관성, 반성을 모두 적기 때문에 연구에 대한 총체적 기록이며 질적 연구자에게 필수적인 연구이다[19]. 본 연구에서는 매 수업시간마다 학생들의 활동현상을 관찰하고 기록함으로써 현장 상황을 분석하는데 사용하였다.

인터뷰는 연구 참여자로부터 자신의 관점을 표현하도록 유도하는 목적있는 대화를 의미하며, 자연적으로 일어나는 행동을 기록하는 관찰법과 비교해 보았을 때 참여자와의 대화를 통해 그들의 행동을 유발시킨다는 점에서 심층적 정보와 이해 획득을 위한 적극적인 연구방법이라고 할 수 있다[19]. 본 연구에서는 마지막 수업을 마친 후에 수업에 참여한 학생 중 임의로 5명을 선정하여 심층 인터뷰를 실시하였다.

3.3 연구 설계

학생들은 정해진 시간과 실습 공간에 모여서 스크래치 도구를 사용하여 총 7회로 구성된 프로그래밍 실습에 참여하였다. 매 수업시간은 먼저 강사의 도구 설명과 적용 사례에 대한 설명이 1시간 정도 있고 이후 2시간 정도 주어진 과제를 스스로 실습해 보는 시간을 갖도록 총 3시간으로 구성하였다.

실습내용의 구성은 <표 1>과 같다.

<표 1> 실습 내용 구성

차시	내용	설문
1강	Part 1. 고양이와 친해지기 Part 2. 스크래치 화면과 생각 블록 Part 3. 닌자 고양이의 순간 이동술 Part 4. Glad to meet you	설문 1차
2강	Part 5. 마우스를 따라 다니는 유령 Part 6. 피아노 만들기 Part 7. 도형 변신시키기	설문 2차
3강	Part 8. 변수 알아보기 Part 9. 리모컨으로 움직이기 Part 15. 쥐를 잡자, 쥐를 잡자~!	설문 3차
4강	Part 20. 미로게임 1 Part 21. 미로게임 2	설문 4차
5강	Part 16. 물고기 잡기 1 Part 10. 원숭이 바나나 받기 1	설문 5차
6강	Part 12. 내 프로젝트 구상하기	-
7강	Part 12. 내 프로젝트 발표하기	설문 6차

매 수업시간이 끝날 때마다 해당 수업에 대한 설문조사를 시행하였으며, 주요 설문 문항은 <표 2>와 같다.

<표 2> 연구에 사용된 주요 설문 내용

본 연구에 사용된 주요 문항 내용	적용 설문
스크래치는 재미있나요?	모든 설문
나는 프로그래밍을 잘 한다고 생각하나요?	모든 설문
스크래치 사용법은 쉬운가요? (혹은 오늘 배운 내용은 쉬웠나요?)	모든 설문
스크래치를 배운 것이 나에게 도움이 되었나요?	마지막 설문
스크래치를 더 배우고 싶은가요?	마지막 설문
스크래치는 어떤 도구라고 생각하나요?	마지막 설문
스크래치를 배우면 어떤 능력이 좋아질까요?	마지막 설문

설문과는 별도로 매 수업시간 관찰일지를 작성하였으며, 수업의 모든 차시가 끝난 후에는 수업에 참여한 학생 중 5명을 선발하여 심층 인터뷰를 실시하였다.

4. 연구 결과

4.1 프로그래밍 흥미도 변화

학습기간동안 스크래치에 대해 느끼는 흥미정도의 변화를 살펴보면 수업시간을 거듭할수록 조금씩 흥미가 떨어지는 것으로 나타났으며 난이도가 높아질수록 흥미는 최저수준으로 낮아지다가 개별 프로젝트를 끝낸 후에 흥미가 급상승하는 것으로 나타났다.

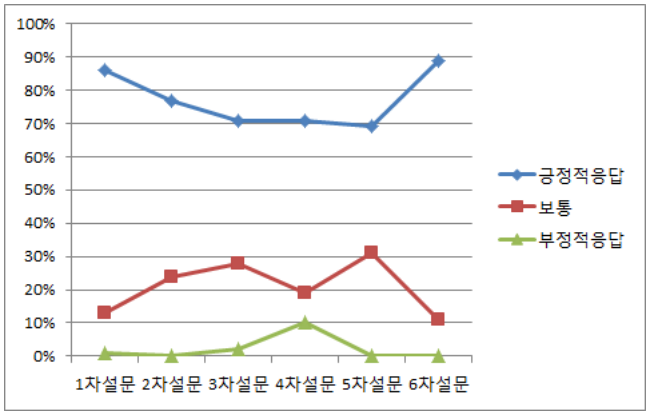
처음 스크래치를 실행해본 학생들은 스크래치가 재미있는가에 대한 설문에 '매우 그렇다'가 34%, '그렇다'가 52%로 스크래치를 재미있다고 긍정적으로 느낀 학생이 86%에 이른 것으로 나타났다. 그러나 수업횟수를 거듭하면서 이 수치는 점점 낮아지는 현상을 보였다. 이는 관찰일지에서도 나타났는데, 학생들의 반응이 소극적으로 변하고 집중하는 모습이 다소 줄어드는 현상을 보였다. 인터뷰에서도 같은 내용을 들을 수 있었는데, 처음에는 쉽고 신기해서 잘 따라하다가 점점 어려워지니까 힘들어졌다고 답하였다. 이는 초반에 느꼈던 새로운 표현도구에 대한 신기함이 점점 줄어들고 프로그래밍 과정에서 느끼는 논리적 사고에 대한 어려움과 피로감이 증가되면서 나타난 현상으로 분석된다.

첫 강의에서 '재미있다'고 답한 학생들은 점점 '매우 재미있다'로 이동하거나 '보통이다'로 이동하는 양극화 현상을 보이다가 스크래치의 주요 함수를 다 배운 시점에서 양극화가 극대화 되었다. 그러나 이후 수업에서는 고급기능이 추가되고 약간의 완성도를 요구하는 미니게임 수준의 예제를 팀별로 문제를 해결하도록 하는 수업이 진행되면서 부터는 '매우 재미있다'라고 답한 학생수가 현저히 줄어들고 '재미없다'라고 답한 학생들도 10%나 되는 현상을 보였다. 이는 고급기능에 대한 어려움이 흥미가 높은 학생이나 낮은 학생 모두에게 부담으로 다가왔음을 알 수 있으며, 아직 프로그래밍이 익숙하지 않은 시점에서 팀단위로 과제를 수행해야 하는 과정에서 이중 부담이 발생하였던 것으로 분석된다. 이 시점에서는 점점 벌어지는 자신과 타인간의 실력과 흥미의 격차가 팀내의 짝을 통해 확실히 인식되면서 포기자가 발생하기도 했다.

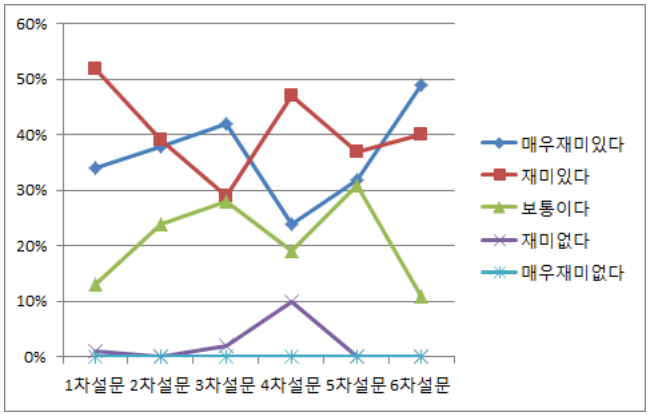
그러나 기말과제로 개인 프로젝트를 마친 후에는 스크래치가 '매우 재미있다'라고 응답한 학생이 49%에 이르고, 재미있다고 답한 학생까지 합하면 스크래치를 재미있다고 긍정적으로 느낀 학생은 89%에 이른 것으로 나타났으며, 전체 수업과정 중에 조사된 가장 높은 수치이다.

이는 김미량(2002)이 연구에서 컴퓨터 프로그래밍 교육에 효과적인 방법으로 프로젝트와 PBL을 제안한 것[20]과 김병욱 외(2010)가 프로그래밍 수업에서 PBL이 특히 학습동기 향상에 효과적인 학습모형임을 밝힌 것[21]에 근거해 볼 때, 자신이 계획하고 설계한 PBL 형태의 개인별 프로젝트 수업이 학생들의 흥미를 크게 상승시킨 요인으로 분석된다.

결론적으로 수업을 거듭하면서 난이도가 증가함에 따라 점점 낮아진 재미와 흥미가 개별과제를 수행하고 나서 느끼는 성취감으로 인해 급상승한 것으로 분석된다.



<그림 1> 수업별 흥미도에 대한 긍·부정의 변화



<그림 2> 수업별 흥미도에 대한 세부 응답별 변화

4.2 도구 용이성 변화

비전공자를 위한 도구로서 스크래치 사용이 쉬운가를 알아보기 위한 설문 문항에서는 수업 횟수를 거듭할수록 오히려 다소 어렵다고 느낀 것으로 나타났다.

첫 수업에서 스크래치의 사용이 '매우 쉽다' 혹은 '쉽다'라고 답한 학생은 각각 11%와 31%로 이를 합하면 절반에 가까운 42% 정도의 학생들이 스크래치의 사용이 쉽다고 느꼈다. 반면 '어렵다' 혹은 '매우 어렵다'라고 답한 학생은 각각 20%와 1% 정도였다. 그러나 마지막 수업에서는 스크래치의 사용이 '매우 쉽다' 혹은 '쉽다'라고 답한 학생은 각각 8%와 25%에 그쳤으며 '매우 어렵다'라고 답한 학생은 없었으나 '어렵다'라고 답한 학생은 22%정도로 나타났다.

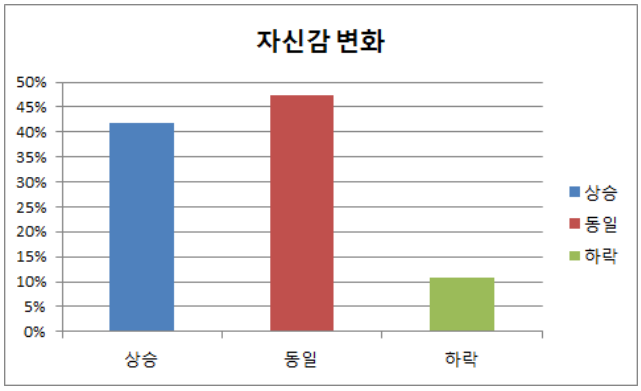
이는 스크래치 자체의 사용법이 어렵고 복잡해서 나타난 현상이라고 볼 수는 없으며[18], 스크래치를 활용하여 해결할 수 있는 문제의 범위가 확대되고 문제의 난이도가 높아지면서 스크래치의 기본 제공 기능 외에 창의적으로 응용해서 사용하는 것 자체가 어렵다고 느낀 것으로 분석되었다.

이러한 현상은 관찰일지를 통해서도 알 수 있었는데, 학생들이 명령어 블록을 블록 팔래트에서 바로 찾아 사용할 수 있는 수준의 난이도에서는 스크래치가 도구로서 매우 쉽다고 느낀 반면, 문제의 난이도가 높아지면서 필요한 명령어 블록이 블록 팔래트에서 바로 제공되지 않고 여러 개의 명령 블록을 복합적으로 응용하여 새로운 명령어 블록 결합체를 만들어내야 하는 수준에 이르거나 방송하기 기능을 사용하여 스프라이트 간의 통신을 통한 상호작용을 해야 하는 수준에 이르게 되자 질문이나 어려움을 호소하는 횟수가 크게 증가하였다.

Rist(1995)에 의하면 프로그램을 작성할 때 비슷한 설계 전략을 보고 프로그램을 설계하게 되는데, 초보 프로그래머의 경우 필요한 프로그램 설계 전략과 비슷한 지식이 없으면 새롭게 개발해야 하는 것이 매우 어렵게 느껴지게 된다[22]. 따라서 과제의 난이도가 높아 이전에 연습했던 전략만으로 해결이 어렵거나, 명령어 블록이 직관적으로 사용 가능한 형태로 제공되지 않은 상황이 발생하면 초보 프로그래머는 창의적으로 문제를 해결하는 데 겪는 어려움을 사고력의 문제 뿐 아니라 도구 사용이 어려워서라고 생각하게 되는 것으로 분석되었다.

4.3 프로그래밍 자신감 변화

프로그래밍에 대한 자신감의 변화를 살펴보면 초반에 가졌던 자신감의 상태가 수업 끝까지 유지되는 경우가 대부분이었으나 초반에 가졌던 자신감에 비해 수업 마지막에 자신감이 상승하는 경우도 42%에 이른 것으로 나타났다.



<그림 3> 자신감의 상승-하락 비율

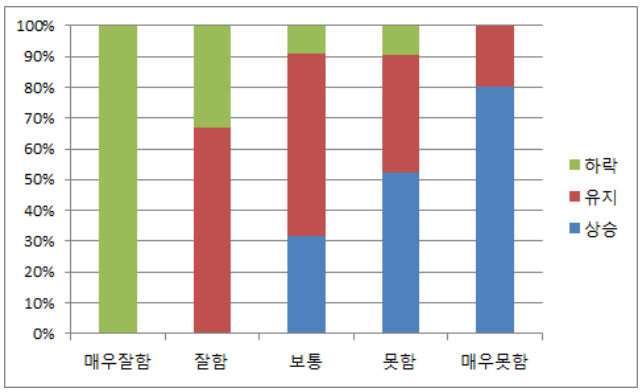
응답내용별로 살펴보면 초반에 '매우 잘한다' 혹은 '잘한다' 라고 프로그래밍에 대해 강한 자신감을 표현했던 학생들은 대부분 학기말에 자신감이 하락한 반면, 초반에 '못한다' 혹은 '매우 못한다' 라고 프로그래밍에 자신감이 없었던 학생들의 상당수는 마지막 수업에서 자신감이 상승하는 현상을 보였다.

첫 수업에서 '잘한다'라고 답한 학생들 중 실습 마지막 수업에서도 동일하게 '잘한다'라고 답한 학생은 50%로 나타났는데, 초반에 자신감을 보였던 학생들이 끝까지 자신감을 가지고 임하는 확률이 절반인 만큼 나머지 50%는 난이도가 높아짐에 따라 자신감을 잃는 현상을 보였다.

첫 수업에서 '못한다' 혹은 '매우 못한다'라고 자신감을 낮게 표했던 학생들 중 41.6%가 마지막 수업에서 '잘한다' 혹은 '매우 잘한다'라고 답하며

변화된 높은 자신감을 보였다.

첫 수업에서 '보통이다'라고 표시했던 학생들의 경우에는 20%가 마지막 수업에서 자신감이 상승한 것으로 나타났다.



<그림 4> 첫 수업 답변별 상승-하락 비율

인터뷰에서도 같은 내용을 확인할 수 있었는데, 이는 수업초반에 단순히 몇 번의 클릭만으로 문제가 해결되는 현상을 보고 강한 자신감을 보였던 학생들이 난이도가 높아짐에 따라 자신감을 상실하게 된 것으로 분석되었다. 이와 반대로, 수업 초반에는 처음으로 겪는 환경에 대한 낯설음으로 인해 자신감 없음을 표현했던 학생들이 수업을 거듭하면서 도구사용에 익숙해지고 스스로 통제가능한 상황에 이르게 되자 자신감이 상승한 것으로 분석되었다.

4.4 프로그래밍 숙련도 변화

프로그래밍 숙련도의 변화를 살펴보기 위해 매 수업시간 프로그래밍 과정을 관찰한 결과 도구의 사용이 익숙해지면서 명령어 블록 선택의 속도가 빨라지고 프로그래밍 절차가 효율적인 형태로 진화하는 현상을 보였다.

비록 스크래치 프로그램이 요철형 블록놀이 형태로 구성되어 있고 직관적으로 해석하기 쉽도록 만들어졌다고는 하지만 프로그래밍을 처음 해보는 학생의 경우 이러한 환경은 너무나 낯설고 어렵게 느껴진다. 이는 지금까지 주로 사용해오던 사고영역과는 조금 다른 형태의 사고력을 요하기 때문인데, 새로운 사고력을 계발해야 함과 동시에 익숙하지 않은 도구를 사용해야 하는 상황이 초보 학습자에게 큰 부담감으로 나타날 수 있다. 그러나 반복적 학습을 통해 도구의 사용이 익숙해짐에 따라 도구 사용에 대한 부담이 줄어들게 되고 명령어에 대한 이해가 높아지면서 본격적으로 사고력 계발에 돌입하게 된다.

처음에는 명령어에 대한 이해가 부족하기 때문에 과제해결을 위해 한 번에 바로 필요한 명령어를 선택하는 학생은 거의 없었다. 다수의 학생들은 명령어 선택에 상당히 주저하는 모습을 보였으며 모범 예시 답안을 본 이후에 최대한 같은 방식으로 따라하려는 경향을 보였다.

이러한 현상은 다른 연구에서도 나타나는 것으로 Piteira(2013)에 의하면 프로그래밍을 배우는데 있어서 학생들은 학습자료 면에서 예제 코드를 가장 유용하게 느끼는 것으로 나타났다[12].

또 다른 현상으로는 아직 명령어 사용법에 대한 이해가 부족하기 때문에 필수 명령어를 선택하는 것에 성공했다고 할지라도 잘못 사용하여 결국 문제를 해결하지 못하는 경우도 있었다.

이해부족으로 인한 시행착오 현상은 다른 연구에서도 알려져 있다. 최정원(2014)은 코드닷오알지를 이용한 프로그래밍 교육에서 초보 학습자가 겪는 어려움에 대해 순차, 반복, 조건, 이벤트의 4가지 프로그래밍 개념으로 구분하여 분석하였다. 여기서 초보 학습자에게 나타나는 공통적인 어려움의 원인으로는 프로그래밍 개념 부족으로 인한 오개념 형성과 시행착오인 것으로 나타났다[23].

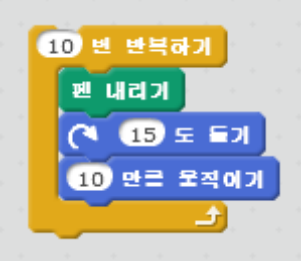
그러나 명령어에 대한 이해가 높아지고 도구사용에 익숙해짐에 따라 명령어를 익히기 위한 탐색의 시간은 줄어들고 문제해결을 위한 사고의 시간이 더 길어지게 되었다.

관찰일지에 따르면 사고과정에 더 많은 고민과 시간이 소요되면서 자연스럽게 프로그램 작성에도 변화가 함께 보이기 시작했다.

예를 들어, "15도로 회전하며 10만큼씩 선을 그리는 행동을 10번 반복하시오"라는 지시문에 대해 프로그래밍을 배운지 2주밖에 되지 않았을 때에는 거의 모든 학생들이 <그림 5>와 같이 작성하였다. 그러나 4주가 지나면서 명령어 사용에 익숙해지게 되고 문제해결에만 집중할 수 있게 되자 <그림 6>과 같이 작성하는 학생들이 늘어나

기 시작했다.

이러한 현상을 통해 프로그래밍이 더 이상 도구사용을 익히는 수준에서 벗어나 사실상 문제해결을 위한 사고과정에 온전히 사용되기 까지는 최소 4주(혹은 4회) 이상의 시간이 걸렸다.



<그림 5> 프로그래밍 수업 2주차

<그림 6> 프로그래밍 수업 4주차

또한, 프로그래밍 초기에는 동일한 시점에 반복적으로 일어나는 두 가지 모듈에 대해 각각을 조건문으로 감싸서 두 개의 조건문 모듈을 작성하는 것이 일반적이었으나 프로그래밍에 익숙해지면 이 전체를 하나의 조건문으로 감싸서 두 가지 모듈을 한 번에 수행하도록 코드 작성이 간결해지는 현상을 보였다.

최현종(2011)이 정리한 초보 프로그래머의 일반적인 특징에 의하면 초보 프로그래머는 프로그램을 이해하는 정신 모델(mental model)이 부족하다고 하였다[9]. D. Storeya 외(1999)가 제안한 정신 모델은 프로그래머가 실제 실행하는 방식대로 변수와 제어의 흐름을 머릿속에 생각하는 것을 말하는데, 초보 학습자에게는 이것이 부족하여 의도하지 않았던 방식으로 실행되는 프로그램을 만나게 될 수 있다[24]. 그러나 프로그래밍이 능숙해지게 되면 정신 모델이 발달하게 되어 제어의 흐름을 실행해보지 않아도 머릿속에서 구현해 볼 수 있게 되고 결과적으로 더 복잡하고 어려운 문제도 해결할 수 있게 된다.

이렇게 도구사용이 더 이상 문제가 되지 않고 문제해결을 위한 사고력에만 집중하게 되는 순간부터는 더욱 자유롭고 다양한 사고가 가능해지는데 특히 자동화 부분에 있어서 코딩이 더욱 효율적인 형태를 갖게 된다.

이를 컴퓨팅 사고력과 연결 지어 생각해보면 문제를 작은 단위로 분해하고 순차적으로 나열하기 위한 사고력은 오류가 발생할 경우 결과적으로 문제가 해결되지 않기 때문에 프로그래밍을 처음 배우기 시작할 때부터 요구되는 사고력이라고 할 수 있으며, 같은 결과를 보이더라도 더욱 효율적으로 동작하도록 자동화할 수 있는 능력은 도구사용이 익숙해진 이후부터 발현되기 시작하는 것으로 볼 수 있다.

4.5 짝 프로그래밍

스크래치 사용에 어느 정도 익숙해질 무렵인 학기 중반 이후에는 주제를 주고 짝을 이루어 함께 아이디어를 내서 창의적으로 만들어 보도록 하였다. 이 때, 짝 간의 프로그래밍 숙련도의 차이정도가 협업과정에 영향을 미칠 것으로 예상하였으나 프로그래밍에 아직 미숙한 상황에서는 숙련도의 차이와 상관없이 자신이 이해한 방식으로만 문제를 해결하고자 하는 성향이 두드러지게 나타나 협업이 잘 이루어지지 않았다.

프로그래밍 숙련도가 높은 학생과 낮은 학생이 짝을 이룬 경우에는 프로그래밍 숙련도가 낮은 학생이 일방적으로 높은 학생의 결정에 따라가는 현상을 보였다. 이러한 조합에서는 숙련도가 낮은 학생이 프로그래밍 구성은 물론 새로운 아이디어 제시에도 어려움을 보였으며 심리적 위화감으로 인해 더욱 수동적인 태도를 보였다.

프로그래밍 숙련도가 비슷한 학생으로 구성된 팀의 경우에는 각자 새로운 아이디어를 제시하고 이를 구현하는 방법에 대해서 활발히 토의하는 현상을 보였으나 작업 전에 논의한 내용보다는 실제 코딩을 하면서 추가되는 아이디어가 더 많이 나오는 것을 알 수 있었다. 이는 아직 프로그래밍에 능숙하지 않기 때문에, 작업 전 기획단계에서 결과물의 기능 전체를 구상할 수 있는 능력

이 아직 부족하고 코딩을 하는 과정에서 새롭게 알게 된 기능들을 통해 아이디어가 추가되는 방식으로 프로그래밍이 진행되기 때문이다.

아이디어 제시에는 이와 같이 활발한 모습을 보였으나 실제로 코딩하는 과정에서는 협업에 어려움을 보였다. 이는 프로그래밍 실력이 아직 높은 수준이 아니므로 해결 가능한 아이디어가 매우 한정적이기 때문에 타인과 협업하는 능력이 부족하고 자신이 할 수 있는 기량의 범위 내에서 찾아낸 해결방법을 강하게 고수하고자 하는 현상이 두드러졌다. 따라서 서로의 주장과 고집이 강해 오히려 협업이 안 되고 각자가 생각한 기능을 개별적으로 개발해서 최종결과물에 합치려는 성향을 보이기도 하였다.

5. 결론

프로그래밍 교육의 중요성에 대한 인식이 커짐에 따라 세계 여러 나라의 교육 기관에서는 보편적 교육으로서의 프로그래밍 교육에 대한 관심이 높아지게 되었다. 특히 인문계열 학생들이 배우는 프로그래밍 수업은 전문가 양성을 위한 프로그래밍 교육과는 차별화된 교육방법이 필요하기 때문에 이들이 겪는 어려움에 대한 면밀한 관찰이 필요한 시점에 있다.

이를 위해 본 연구에서는 학생들이 도구사용법에 지나치게 종속되지 않도록 직관적으로 코딩이 가능한 이미지 형태의 프로그래밍 언어인 스크래치를 사용하였다.

연구 내용으로는 수업횟수를 거듭할수록 학생들의 프로그래밍에 대한 흥미도, 자신감, 숙련도가 어떻게 변하는 지와 팀 내 숙련도의 차이정도가 팀 활동에 미치는 영향에 대해 설문과 관찰, 인터뷰 등을 통해 종합적으로 분석하였다.

그 결과, 프로그래밍의 흥미도 변화에서는 수업시간을 거듭할수록 조금씩 떨어지는 것으로 나타났으며 난이도가 높아질수록 흥미는 최저수준으로 낮아지다가 개별 프로젝트를 끝낸 후에 흥미가 급상승하는 것으로 나타났다. 이는 고도의 집중과 두뇌회전을 요구하는 프로그래밍의 과정은 힘들지만 이를 해결한 후에 느끼는 쾌감이 흥미를 더욱 크게 향상시키는 것으로 나타났다.

프로그래밍의 자신감 변화에서는 수업초반에 프로그래밍에 강한 자신감을 표현했던 학생들은 대부분 학기말에 자신감이 하락한 반면, 수업초반에 프로그래밍에 자신감이 없었던 학생들의 상당수는 마지막 수업에서 자신감이 상승하는 현상을 보였다. 이는 수업초반에 단순히 몇 번의 클릭만으로 문제가 해결되는 현상을 보고 강한 자신감을 보였던 학생들이 난이도가 높아짐에 따라 자신감이 위축되는 현상을 보였다. 이와 반대로, 수업 초반에는 처음으로 겪는 환경에 대한 낯설음으로 인해 자신감 없음을 표현했던 학생들이 수업을 거듭하면서 도구사용에 익숙해지고 스스로 통제가능한 상황에 이르게 되자 자신감이 상승하는 현상을 보였다.

프로그래밍의 숙련도 변화를 관찰한 결과로는 수업을 거듭할수록 학생들의 도구 사용이 익숙해지면서 명령어 블록 선택의 속도가 빨라지고 잘못 선택하는 횟수도 줄어들었으며 무엇보다도 프로그래밍 절차가 효율적인 형태로 진화하는 현상을 보였다.

마지막으로 짝 프로그래밍의 수행과정을 관찰해본 결과, 프로그래밍에 아직 미숙한 상황에서는 숙련도의 차이와 상관없이 자신이 이해한 방식으로만 문제를 해결하고자 하는 성향이 두드러지게 나타나 협업이 잘 이루어지지 않음을 알 수 있었다.

본 연구를 통해 알게 된 인문계열 학생들이 처음 프로그래밍 수업에서 겪는 흥미도, 자신감, 숙련도의 변화와 짝 프로그래밍에서 나타나는 현상에 대한 기술은 근거이론(grounded theory) 단계로 제안하는 것으로 일반화를 위한 연구가 후속되어야 할 것이다. 향후 계속되는 연구를 통해 도출되는 연구결과들은 보편적 프로그래밍 교육을 위한 지침과 교수학습방법 설계에 도움을 줄 수 있을 것으로 기대된다.

참 고 문 헌

[1] 김석원,박강민,강송희(2014). **과학연구에서 SW활용 사례**. 소프트웨어정책연구소. 이슈리포트 2014-008.

[2] 한국컴퓨터교육학회(2014). **문이과 통합형 개정 교육과정에 소프트웨어 교육반영을 위한 공개토론회**. 2014.07.03. 전경련회관.

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

[4] College Board (2014). *The AP Computer Science Principles Curriculum Framework 2016-2017.*

[5] Andrea diSessa. (2000). *Changing Minds: Computers, Learning, and Literacy.* MIT Press.

[6] 김수환, 한선관, 김현철(2010). Computational Literacy 교육에서 프로그래밍 능력과 학습자 특성에 관한 연구. **한국컴퓨터교육학회 논문지** 13(2). 15-23

[7] 이영준 외(2014). **초중등 단계 Computational Thinking 도입을 위한 기초연구**. 한국과학창의재단.

[8] 김현철, 성정숙, 김민자(2014). **초·중등 소프트웨어 교육 강화를 위한 연구**. 한국정보통신산업진흥원.

[9] 최현종(2011). 대학 프로그래밍 강좌를 위한 프로그래밍 교육 프레임워크. **한국컴퓨터교육학회논문지** 14(1), 69-79.

[10] Dehnadi, Bornat(2006). The camel has two humps. *PPIG 2006.*

[11] Douglas Rushkoff (2010). *Program or be programmed: Ten Commands for a digital age.* OR Books, New York.

[12] Piteira, M., & Costa, C. (2013, 7). *Learning computer programming: study of difficulties in learning programming.* In Proceedings of the 2013 International Conference on Information Systems and Design of Communication, Lisboa. ACM.

[13] 신수범(2014). **교육용 프로그래밍 언어 선택 전략**. 한국교육학술정보원. 기타자료 PM 2013-4

[14] Scratch Community statistics at a glance http://scratch.mit.edu/statistics/ (2014. 11. 12 검색)

[15] 성정숙, 김현철(2015). 국외 컴퓨터 교육과정 변화 분석 연구. **한국컴퓨터교육학회논문지**. 18(1).

[16] National Research Council. (2009). *Report of a Workshop on The Scope and Nature of Computational Thinking.*

[17] National Research Council. (2010). *Report of a Workshop on The Scope and Nature of Computational Thinking.*

[18] 김수환, 이원규, 김현철(2009). 개정된 정보교육과정에서 교육용프로그래밍언어의 교육적 적용방안. **한국컴퓨터교육학회논문지** 12(2), 23-31.

[19] 김영천(2012). 질적연구방법론1: Bricoleur 제2판. 아카데미프레스.

[20] 김미량(2002). 컴퓨터 프로그래밍 교육에 적용 가능한 효과적 교수 방법의 탐색적 제안. **한국컴퓨터교육학회논문지** 5(3), 1-9.

[21] 김병욱 외(2010). PBL 기반 프로그래밍 수업에서 학습양식에 따른 학습 동기 차이 분석을 통한 시사점 도출. **한국컴퓨터교육학회 논문지** 13(5), 15-27.

[22] Rist, R.S. (1995). Program structure and design. Cognitive Science, 19, 507-562.

[23] 최정원, 이영준(2014). 프로그래밍 학습에서 학습자의 어려움 분석. **한국컴퓨터교육학회 논문지**. 17(5). 89-98.

[24] D. Storeya, F. D. Fracchiaa and H. A. Mullerb (1999). Cognitive design elements to support the construction of a mental model during software exploration, Journal of Systems and Software, 44(3), 171-185.

성 정 숙

2001 한동대학교
전산전자공학부(공학사)
2004 연세대학교
교육학과(전자계산교육학석사)
2012~현재 고려대학교 컴퓨터교육학과 박사과정
관심분야: 컴퓨터 교육과정, SW교육,
컴퓨팅 사고력, 컴퓨팅 리터러시
E-Mail: workwithu@gmail.com

김 수 환

1999 인천교육대학교(교육학학사)
2006 경인교육대학교
컴퓨터교육과(교육학석사)
2011 고려대학교
컴퓨터교육과(이학박사)
2013~2014 경인교육대학교 겸임교수
2014~현재 총신대학교 교양교직과 조교수
관심분야: 컴퓨터교육, EPL, 컴퓨터적사고, CSCL,
Computational Literacy
E-Mail: skim@csu.ac.kr

김 현 철

1988 고려대학교
전산과학과(이학사)
1990 Univ. of Misouri-Rola
(전산학 석사)
1998 Univ, of Florida (전산정보학 박사)
1999~현재 고려대학교 사범대학 컴퓨터교육과 교수
2014~현재 고려대학교 정보대학 컴퓨터학과 교수
관심분야: 컴퓨터교육, 기계학습
E-Mail: harykim@korea.ac.kr