과학영재 고등학교 도구교과로서의 프로그래밍 교육과정 내용요소 설계

김종혜†

요 약

본 연구에서 프로그래밍 교육과정의 내용 요소는 정보과학 영재가 아닌 과학영재학생들을 위한 도구교과로 설계되었다. 먼저, 도구교과로써의 프로그래밍 교육과정은 2학기 동안의 수업 결과를 바탕으로, 프로그래밍의 이해, 객체지향 프로그래밍, 시뮬레이션 프로그래밍의 3영역으로 구성하여 내용요소를 설계하였다. 그 후, 과학영재 고등학교 학생들에게 수학, 과학 수업이나 연구에서 활용 가능 여부 확인을 위해, 한 학기동안 수업 및 프로젝트과제, 문제해결과제를 수행하였다. 연구 결과를 통해, 학생들이 Computational thinking 기반의 문제해결능력 뿐 아니라 수학, 과학 분야에서의 수치 해석 및 시뮬레이션 프로그램 개발 역량이 향상됨을 알 수 있었다. 또한, 학생들은 프로그래밍 학습이 과학, 수학 공부나 연구를 하는데 필요한 도구교과라는 생각으로 바뀐 것을 알 수 있었다. 본 연구결과가 과학영재학교에서 도구교과 성격으로서의 정보 교과의 프로그래밍 교육과정을 설계하는데 가이드라인을 제시할 수 있을 것이라 기대한다.

주제어 : 과학영재학생, 프로그래밍 교육과정, 도구교과

Designing Content Elements of the Programming Curriculum as a Instrumental Subject for Gifted Science High School

JongHye Kim†

ABSTRACT

In this study, contents of programming curriculum were designed as instrumental subjects for scientifically gifted students, not for IT gifted ones. Firstly, the programming curriculum consisted of 3 sections; Programming Understanding, Object-Oriented Programming, and Simulation Programming as a result of two semesters. Then, the lectures including project-based and problem-solving tasks were given to scientifically gifted students in a high school during one semester to verify whether they could apply the contents to studies and researches in math or science or not. As a result of this study, the students could improve numerical analysis and simulation program development capabilities in math or science as well as the problem-solving ability based on computational thinking. Moreover, it was proved that the students changed their perception about programing learning. They started to think that programing learning was necessary to studies and researches in math or science. The results of this study propose guideline to design programming curriculum as instrumental subjects for scientifically gifted students.

Keywords : Scientifically Gifted Students, Programming Curriculum, Instrumental Subject

† 정 회 원: 경기과학고등학교(교신저자)
논문접수: 2014년 2월 3일, 심사완료: 2014년 4월 22일, 게재확정: 2014년 5월 26일

1. 서론

2009 개정 정보 교육과정은 '컴퓨터 활용 중심의 컴퓨터 교육'에서 '사고 중심의 정보 교육' 으로의 방향 전환한 2007 개정 교육과정의 목표를 근간으로 수정, 보완되였다[1][2][3][4][5][6]. 특히, 현 사회에서는 과학 뿐 아니라 지리, 수학, 경제 등 정보가 사용되지 않는 분야가 없기 때문에, 2009 개정 중등 정보 교육과정은 정보기술 유창성에 이어 정보과학적 사고(Computational Thinking)을 기반으로 한 문제해결능력 향상을 중점으로 하는 도구 교과로서의 성격을 내포하고 있다[3][4]. 또한, 전 세계적으로 정보 교과를 초등학교 교육과정에부터 필수교과로 지정되고 있는 최근의 추세는 정보교과가 도구 교과로서의 방향성을 가지고 있음을 보여주고 있다. 미국 의회는 초·중등교육에서 정보과학을 필수 교과로 지정하기 위해 2011년부터 '컴퓨터 과학 교육 법령(Computer Science Education Act of 2011)'(H.R. 3014) 을 채택하였으며, 핀란드 또한 IT 조기 교육 방침으로 초등학생 때부터 프로그래밍 기초 기술을 가르치는 교육과정을 도입하겠다고 공표하였다[6][7]. 또한, 영국은 2014년부터 알고리즘 이해와 응용 프로그램은 물론, 문제해결을 위한 프로그래밍 내용을 포함시킨 컴퓨팅(Computing) 과목이라는 이름으로 초, 중등 전 학년에 필수화한다고 공표하였다[8].

이러한 도구교과로서의 정보 교과는 학교 급별, 특성별로 차이를 가져야 될 것이다. 과학영재 고등학교에서의 정보 교과는 정보과학 영재 육성과 더불어 미래의 자연과학, 공학자로서 이론의 검증 및 실험을 수행할 수 없는 극한 상황이나 나노 또는 빅 데이터 처리 수행이 가능한 융합 역량을 갖출 수 있는 교육과정이 수립이 요구된다. 특히 과학영재들이 수학이나 과학의 이론 검증 및 데이터 분석을 통한 예측이 필요할 때에 정보교과는 도구교과로서의 역할을 가질 필요가 있다[9][10][11]. 2013년 노벨화학상이 거대분자의 구조와 화학반응을 예측하는 컴퓨터 시뮬레이션 프로그램을 개발한 화학자들에게 돌아간 것이 그 대표적인 예이다[12]. 이 노벨 화학상은 순수과학에서 개발된 소프트웨어도 새로운 이론이나 실험만큼 중요한 과학적 업적으로 인정받게 됐다는 것을 보여주었다. 또한, 생물정보학(BioInformatics), 전산 물리학(Computational Physics), 계산화학(Computational Chemistry) 등의 연구 분야가 있는 것처럼, 과학영재학생들에게 실험으로 한계를 가지고 있는 자연과학의 문제를 컴퓨터 프로그램을 이용하여 해석, 해결하는 것도 필요한 역량이 되어버렸다.

이에 본 연구에서는 정보과학영재가 아닌 과학영재학생들을 위한 프로그래밍 교육과정으로서의 내용요소를 설계하고, 그 효과성을 검증하고자 한다.

2. 관련연구

2.1 도구교과로서의 정보교과

도구 교과는 다른 교과를 학습하는데 도구로 사용되는 교과이며, 고등사고능력을 길러주는데 기반이 되는 교과를 말한다. 정보 교과는 도구적 성격을 가지고 있다는 실천적, 실험적 연구 뿐 아니라 교육적 패러다임 연구들을 통해 도구 교과로서의 근거들을 제시하고 있다[13][14][15][16].

Wing(2006)은 도구 교과의 관점은 읽고, 쓰고, 셈하는 3R을 핵심으로 하는 것 뿐 아니라, 정보과학적 사고(Computational Thinking)가 필요하다는 것을 제시하고, 미래의 모든 과학과 공학이 컴퓨팅에 기반이 된다는 다양한 근거들을 제시하였다[13]. 김자미(2010)는 정보 교육을 알고리즘적 사고를 바탕으로 발견해 나가는 컴퓨터 과학의 지식과 원리에 대한 이해, 프로그래밍 과정에서 얻게 되는 고등 수준의 인지능력의 발달 등을 포함한 본질론적 정당화와, 해당 학문 분야 뿐 아니라 수학이나 과학 등 다른 학문으로의 전이를 통해 기대할 수 있는 도구론적 정당화로 구분하였다[14]. 이영준 외(2008)는 수학이나 과학 교과가 읽기, 쓰기와 더불어 모든 사람들이 반드시 배워야 하는 기초 학문으로 간주되는 것처럼 정보 교과 역시 21세기를 살아가는 모든 사람들에게 필요한 기초적인 원리와 기술을 습득하기 위한 필수적인 학문 영역이라고 주장하였다[15]. Denning(2007)은 컴퓨팅이 더 이상 응용 기술이나 컴퓨터라는

도구에만 치중한 학문분야가 아니라, 모든 사람이 필수적으로 배워야 하는 기초 과학이라는 새로운 패러다임을 제시하고 있다[16]. 또한 컴퓨터 과학이 수학이나 공학의 영향력 아래 생산된 학문이지만, 반대로 컴퓨터 과학은 수학과 많은 공학의 원리에 강한 영향력을 행사하고 있을 뿐 아니라 필수 불가결한 요소가 되고 있다는 것을 제시하고 있다.

이에 정보 교과는 21세기를 살아가는 모든 사람에게 필요한 기본적인 학문이며 이를 통해 고등 수준의 인지능력을 발달시킬 수 있고, 도구로서 다른 학문 분야의 효율성을 높이는 과목일 뿐만 아니라 독립된 학문으로서의 성격도 지니고 있다고 볼 수 있다.

2.2 연구역량 수행을 위한 프로그래밍 언어

정보 교과는 사회, 경제, 정치 분야 등의 정량적, 정성적 가치 창출 뿐 아니라 자연과학, 공학에서부터 인문학까지 IT 기반의 분석, 처리가 요구되고 있다는 점에 기반을 두고 교육과정이 설계되어야 한다. 이에 과학영재 고등학교에서의 정보 교과 목표는 정보과학 영재 육성과 더불어 연구 이론의 설계, 검증 및 실험을 수행할 수 없는 극한 상황, 빅 데이터 처리 등 융합 역량을 갖출 수 있는 교육과정이 수립이 요구된다.

특히, 정보 교과 중 프로그래밍 교육은 IT 뿐 아니라 공학, 자연과학의 연구를 수행하는데 있어서 중요한 교육내용이다. 이러한 프로그래밍 교육은 대학 학부 과정에서 컴퓨터 공학과 및 정보통신 학부를 제외한 학과에서도 중요한 내용요소로 자리 잡고 있다. 정여진(2010)은 총 16개 대학에서 컴퓨터공학과 및 정보통신 학부를 제외한 학과들의 모든 과목 강의 계획서를 분석하여 각 과목별로 요구하는 컴퓨터 과학 내용 요소와 ICT 활용 내용 요소를 추출하여 프로그래밍 교육의 중요성을 보여주었다[17]. 16개 대학의 모든 학과(정보통신학부, 컴퓨터 공학과 제외)에서 가장 많이 필요로 하는 내용요소는 프로그래밍 영역이었으며, 가장 많이 필요로 한 언어는 C, Matlab, Python 등의 결과로 나왔다.

대학 뿐 아니라 고등학교 수학, 과학 분야에서도 Python 또는 VPython은 정밀한 데이터 처리나 다양하고 복잡한 시뮬레이션 프로그램을 비교적 쉽게 만드는데 효과적인 언어임이 많은 연구결과들을 통해 증명되었다[18][19][20][21][22].

Sheikh Iqbal Ahamed (2010) 는 고등학교 과학, 수학 교사들에게 수학과 과학 분야에서 정보과학적 사고(Computational Thinking)를 가르치기 위해 Python과 VPython을 이용한 교사 연수를 실시하였다. 교사 연수 결과, Python 과 VPython 이 문법을 쉽게 배울 수 있고, 짧은 프로그램으로도 다양하고 복잡한 시뮬레이션을 만들 수 있기 때문에, 수학 뿐 아니라 물리, 생물, 천문 등의 과학 수업이나 연구에서 이용하기 효과적이라는 결과를 얻었다[18]. 또한, Vicki Allen(2010) 은 고등학교 수학, 화학, 물리 교육과정에서 정보과학적 사고(Computational Thinking)를 하는데 Python, VPython 이 도움이 된다는 근거들을 보여주었다[19]. 예를 들어, 물리 영역에서는 물리적 원칙과 모델을 설명한 후 파이썬을 이용해 시뮬레이션하면서 학생들이 정보과학적 사고의 개념을 적용할 수 있음을 제시하였다. 본 연구에서 설계하고자 하는 과학영재학생들을 위한 프로그래밍 교육과정도 위 연구들과 같이, 물리, 화학 등의 과학 분야를 개념, 실험으로 확인하는 것 이상의 더 큰 데이터를 처리할 수 있고, 복잡하고 어려운 실험도 가능할 수 있는 시뮬레이션 알고리즘 구현 능력을 목표로 하고 있다. 특히, 과학영재학생들이 기존의 내용을 확인하는 수준이 아닌 학생 주도적으로 연구가 이루어질 수 있도록 프로그래밍 교육과정 설계가 요구된다.

전 세계적으로 많이 사용되는 프로그래밍 언어 순위를 확인한 결과, 2007년부터 2013년까지 C언어는 1~2위, Python은 6~8위, Matlab은 16~19위를 보여주었다[23]. 프로그래밍 언어 패러다임에서는 객체지향 언어(Object-oriented language)가 56.2%, 절차적 언어(Procedural language)가 36.7%, 함수형 언어(Functional language)는 3.6%, 논리형 언어(Logical language)가 3.4%의 순위를 보여주었다.

위의 연구결과들을 통해, 본 연구에서는 현재 전 세계적으로 선호도가 높은 프로그래밍 언어 패러다임인 객체지향 언어이면서, 프로그래머가

아닌 사람들도 쉽게 프로그램을 개발해 다른 학문 분야로의 전이가 쉬운 Python(RUR-PLE, VPython 포함)을 선택하여 교육과정 내용요소를 설계하였다.

3. 연구방법

본 연구는 과학영재 고등학교에서 정보과학영재들이 아닌 과학영재학생들을 위한 도구 교과로서의 프로그래밍 교육과정 내용요소를 설계하였다. 또한, 설계한 내용요소를 기반으로 과학영재 고등학생들에게 한 수업과 과제를 통해 수학이나 과학 교과에서 응용이 가능한 시뮬레이션 프로그램을 개발할 수 있는지에 대해 검증하였다.

먼저, 프로그래밍 교육과정의 영역과 내용요소를 1년 동안의(2012학년도 2학기, 2013학년도 1학기) '객체지향 프로그래밍' 과목 수업 리뷰를 바탕으로 다음과 같은 방향으로 설계하였다.

- 파이썬(Python)과 로봇 기반의 교육용 파이썬 프로그래밍 언어인 러플(RUR-PLE)을 이용하여 문제해결방법 이해 및 알고리즘 구현이 이루어지도록 한다.
- 파이썬(Python)을 이용하여 객체지향의 개념과 원리를 이해하고, 객체지향 설계를 기반으로 한 객체지향 프로그램 개발이 이루어지도록 한다.
- 비주얼 기반의 VPython을 이용하여 수치해석 및 과학 시뮬레이션 프로그램 개발이 이루어지도록 한다.

객체지향 프로그래밍 과목 수강 대상자 특징 및 수업 특징은 <표 1>과 같다.

<표 1> 수강대상자 및 수업 특징

<table>
<tr><td>수업 시수</td><td colspan="4">3시간(주)*14주 = 42시간</td></tr>
<tr><td>수강 대상자</td><td colspan="4">• 과학영재 고등학교 3학년 25명(정보과학 심화수업 이수자 제외)
• 1학년 물리, 화학, 생물, 지구, 정보 과학 과목 필수 수강 - 2학년부터 과학 과목 선택 수강
• 1학년 수강한 정보 과학 과목 내용: C Programming(2시간/1주)</td></tr>
<tr><td>수업 내용</td><td colspan="4">프로그래밍의 이해, 객체지향 프로그래밍, 시뮬레이션 프로그래밍</td></tr>
<tr><td>사용 언어</td><td colspan="4">Python(RUR-PLE, VPython)</td></tr>
<tr><td rowspan="3">수행 과제 및 평가</td><td>영역</td><td>프로그래밍의 이해</td><td>객체지향 프로그래밍</td><td>시뮬레이션 프로그래밍</td></tr>
<tr><td>프로젝트</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td>문제해결</td><td>○</td><td></td><td></td></tr>
</table>

본 연구에서는 '객체지향 프로그래밍' 과목 수업을 들은 31명의 학생들 중 정보과학이 전공이 아닌 3학년 25명의 학생들을 대상으로 하였다. 이 학생들은 1학년 때 정보 과학 과목(C programming)을 매주 2시간씩 총 14주를 수업을 받았으나, 2학년 때에는 정보과학 과목을 선택하지 않아 1년 동안 프로그래밍을 한 경험이 없고, 객체지향 프로그래밍에 대한 개념과 경험이 없는 걸로 나타났다.

수업시간은 매주 3시간씩 총 14주로 진행되었으며, 총 4번의 수행평가를 실시하였다. 프로그래밍의 이해, 객체지향 프로그래밍, 시뮬레이션 프로그래밍의 각 영역별로 프로젝트 과제를 공통적으로 부여하였다. 추가적으로, 프로그래밍의 이해 영역에서는 주어진 문제를 알고리즘으로 설계해 구현하는 문제해결 평가를 추가로 수행하였다.

마지막으로, 본 연구를 통해 학생들이 프로그래밍에 대한 인식이 어떻게 변화되었는지 알기 위해, <표 2>와 같이 지적 및 문제해결, 프로그래밍에 대한 자신감에 대한 9개의 5점 평정 척도 문항과 주관식 설문 문항을 설계하였다. 수업을 듣기 학기 초, 학기 후에 자기 설문 조사를 실시하여 평균값을 비교하여 프로그래밍에 대한 인식의 변화를 확인하였다.

<표 2> 자기 설문 문항

Q	자기 설문 문항
1	프로그래밍은 내가 연구하는 영역에서 중요하다고 생각한다.
2	프로그래밍은 내가 공부하거나 연구하는 영역에서 사용하기 쉽다.
3	나는 프로그래밍이 수학 또는 과학 분야를 공부하는 데 도움을 준다고 생각한다.
4	나는 프로그래밍을 더 배우고 싶은 생각을 가지고 있다.
5	나는 클래스와 객체의 개념을 알고 있다.
6	나는 Python 또는 VPython을 이용하여 프로그램을 작성할 수 있다.
7	나는 문제해결전략을 구상하고, 그것을 정보처리시스템에서 효율적으로 수행될 수 있도록 알고리즘을 설계해 구현할 수 있다.
8	나는 프로그래밍을 통해 수치 해석 문제를 해결할 수 있다.
9	나는 시뮬레이션 프로그램을 개발할 수 있다.
10	한 학기동안 수업을 받은 후 느낀점을 자유롭게 서술해 주세요.

4. 연구결과

정보과학영재가 아닌 과학영재 고등학생들을 위한 프로그래밍 교육과정 영역 및 내용요소를 설계하고, 수업을 통해 프로그래밍 교과가 도구교과로서 검증한 결과는 다음과 같다.

4.1 과학영재 고등학생들을 위한 도구교과로서의 프로그래밍 교육과정 영역 및 내용요소 설계

프로그래밍 교육과정의 영역과 내용요소는 1년 동안의 '객체지향 프로그래밍' 과목 수업 리뷰를 바탕으로 다음과 같이 총괄 목표와 영역별 세부 능력, 내용요소로 설계하였다.

먼저, 프로그래밍 교육과정의 총괄 목표는 프로그래밍 원리 및 객체지향 프로그래밍의 개념과 특징을 이해하고, 객체지향 설계와 GUI 프로그래밍을 통해 정보 교과 뿐 아니라 타 교과의 이론 검증 및 실험을 수행할 수 없는 극한 상황이나 빅 데이터 처리를 수행할 수 있는 알고리즘 구현 및 시뮬레이션 프로그램을 작성하는 것으로 설정하였다.

다음으로, 프로그래밍 교육과정을 '프로그래밍의 이해', '객체지향 프로그래밍', '시뮬레이션 프로그래밍'의 총 3개의 영역으로 구분하여 내용요소를 설계하였다. 각 영역의 세부능력과 내용요소는 다음과 같다.

첫째, '프로그래밍의 이해' 영역에서는 Python과 RUR-PLE을 이용하여 프로그래밍의 원리를 이해하고, 주어진 문제를 해결할 수 있는 능력을 향상시킨다. 또한, 수식이나 논리적 사고과정을 비재귀적, 재귀적 알고리즘으로 설계해 구현할 수 있는 능력을 키운다.

둘째, '객체지향 프로그래밍' 영역에서는 Python을 이용하여 객체지향의 원리 및 특징을 이해하고, 문제를 정의하거나 주어진 문제를 객체지향적으로 설계하여 구현하는 능력을 키운다.

셋째, '시뮬레이션 프로그래밍' 영역에서는 Python, VPython을 이용한 GUI 프로그래밍을 통해 시뮬레이션 프로그램을 구현할 수 있는 능력을 키운다. 다양한 수치해석 문제의 그래프를 그리고 문제를 해결하는 능력을 키운다. 수학 뿐 아니라 물리, 천문, 화학, 생물 분야에서 필요로 하는 시뮬레이션 프로그램을 구현할 수 있는 능력을 키운다.

<표 3> 프로그래밍 교육과정 내용요소

대영역	중영역	내용요소	주
프로그래밍의 이해	프로그래밍 언어의 이해	프로그래밍 언어의 이해	1
	함수의 이해	함수의 정의 함수의 호출	2
	제어문의 이해	조건문의 이해 반복문의 이해	
객체지향 프로그래밍	객체지향의 이해	객체지향 개념 객체지향 프로그래밍 특징	1
	객체지향 설계	객체지향 설계 객체지향 설계를 기반으로 한 프로그래밍	3
시뮬레이션 프로그래밍	GUI 프로그래밍	Python 기반의 GUI 프로그래밍 VPython 기반의 GUI 프로그래밍	3
	수치해석 프로그래밍	방정식 문제 해법 구하기 수치적분법 문제 해법 구하기 미분 방정식 문제 해법 구하기	2
	시뮬레이션 프로그래밍	과학(물리, 생물, 화학, 천문, 지구과학) 분야 시뮬레이션 프로그래밍	2

4.2 프로그래밍 교육과정 영역별 과제 및 평가 결과

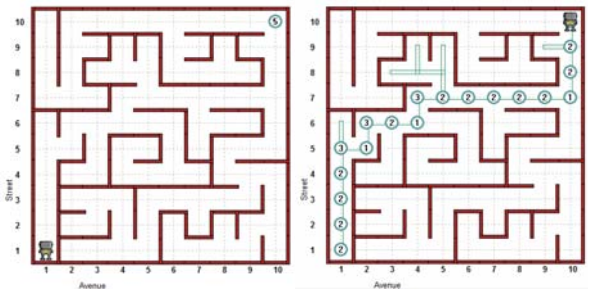
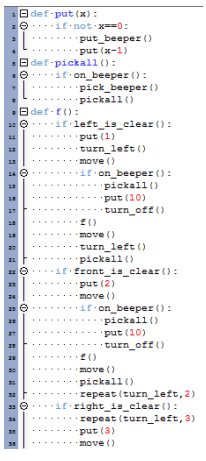
본 연구에서는 설계한 프로그래밍 교육과정 내용요소를 직접실습법, 문제해결법, 프로젝트법을 기반으로 수업 및 평가를 실시하였다. 프로그래밍 교육과정의 각 영역별로 실제 수업에서 수행한 과제 및 평가 결과는 다음과 같다.

4.2.1 '프로그래밍의 이해' 영역 수행 과제 및 평가

'프로그래밍의 이해' 영역은 문제해결평가와 프로젝트 과제를 수행평가로 실시하였다.

먼저, 학생들에게 난이도가 쉽고 어려운 아이디어가 다른 2가지 경우의 프로젝트를 제시하였다. 학생들에게 로봇을 이용해 해결할 수 있는 문제 상황을 정의, 알고리즘을 설계하여 구현하는 것으로, 로봇을 이용해 해결할 수 있는 문제가 어떤 것이 있는지 스스로 생각할 수 있도록 독창성과 유창성을 향상시키는 것이 목표이다. 또한, 자신과 다른 사람들이 정의한 문제들을 함께 풀어보면서 로봇으로 해결할 수 있는 다양한 문제들을 살펴보고, 직접 해결해 볼 수 있도록 하였다.

문제 정의 : 시각장애인 안내 로봇
시각 장애인이 비퍼를 찾아가려고 하는데 시각 장애인이라 길을 알려줄 비퍼가 필요하다. 안내 로봇이 시작지점부터 비퍼까지 가는 길을 비퍼로 나타내어라. 직진을 해야되면 2비퍼, 좌회전을 해야되면 1비퍼, 우회전을 해야되면 3비퍼를 놓는다. 만약 비퍼를 찾으면 도착했다는 것을 알려주기 위해서 10비퍼를 놓는다. 맵에 비퍼가 없으면 시작지점에 18을 놓고 한칸 앞으로 움직인다.

설계한 맵	프로그램 일부
	

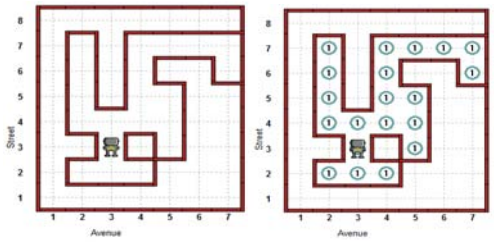
```
def put(x):
    if not x==0:
        put_beeper()
        put(x-1)
def pickall():
    if on_beeper():
        pick_beeper()
        pickall()
def f():
    if left_is_clear():
        put(1)
        turn_left()
        move()
        if on_beeper():
            pickall()
            put(10)
            turn_off()
        f()
        move()
        turn_left()
        pickall()
    if front_is_clear():
        put(2)
        move()
        if on_beeper():
            pickall()
            put(10)
            turn_off()
        f()
        move()
        pickall()
        repeat(turn_left,2)
    if right_is_clear():
        repeat(turn_left,3)
        put(3)
        move()
```

<그림 1> '프로그래밍의 이해' 영역 프로젝트 과제 예

<그림 1>은 학생이 수행한 프로젝트 과제(쉬운 난이도)의 한 예로, 시각장애인 안내 로봇이 목적지까지 가장 최적화된 길을 찾을 수 있는 아이디어를 제시하여 맵을 설계하고 알고리즘을 구현하였다.

두 번째로, 문제해결평가는 학생들이 교사가 주어진 문제를 분석해 알고리즘으로 설계해 구현할 수 있도록 설계하였다. <그림 2>는 문제해결평가 문항의 한 예로, 섬의 모든 지역에 횃불을 놓고 다시 돌아오는 문제이다.

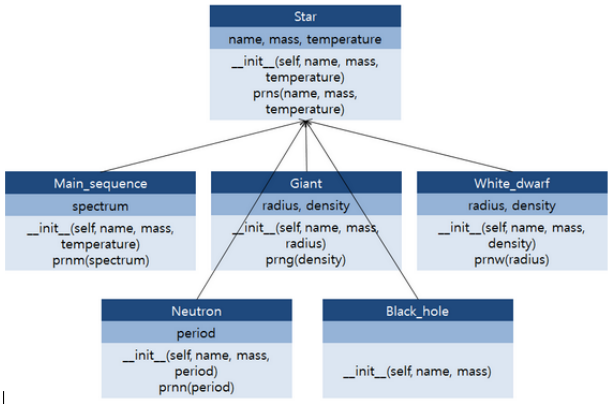
문제 예 : 리봇슨 크루소
리봇슨 크루소는 불의의 사고로 어떤 섬에 불시착하였다. 그는 매우 용맹하여 섬탐사를 나가려고 한다. 어두워도 다닐수 있게 섬의 모든 지역에 횃불을 놓고 다시 집으로 오려고 한다.
우리 모두 리봇슨 크루소를 도와주자.



<그림 2> '프로그래밍의 이해' 영역 문제해결평가 예

과제 예 : 어떤 천문학자는 자신이 그동안 연구해 온 별(Star)들을 객체로 취급하여 분류 및 정리를 하려고 한다. 모든 별은 이름(name), 질량(mass)과 온도(temperature)라는 특성을 기본적으로 가지며, 다만 블랙홀은 온도를 측정하는 것이 불가능하여 온도의 값을 'default'로 지정한다. 별의 종류에는 주계열성(Main_squence), 거성(Giant), 백색 왜성(White_dwarf), 중성자별(Neutron), 블랙홀(Black_hole)이 있다. 주계열성은 스펙트럼(spectrum), 거성과 백색왜성은 반지름(radius)과 밀도(density), 중성자별은 펄스 주기(period)를 부가적인 특징으로 갖는다. 스펙트럼은 별의 온도에 따라 결정되며, O형은 25000K 이상, B형은 10000~25000K, A형은 7000~10000K, F형은 6000~7000K, G형은 5000~6000K, K형은 4000~5000K, M형은4000K 미만의 온도를 갖는다. 거성은 관측을 통하여 반지름을 측정하는 것이 가능하며, 백색 왜성은 역학적 운동을 분석하여 밀도를 알아낼 수 있다. 각각 별의 특징에 반지름이나 밀도 값을 대입하면, 계산을 통하여 다른 값을 출력해준다.(즉, 거성 클래스는 별의 밀도를 계산해주고 백색왜성 클래스는 별의 반지름을 계산해준다.)

Class Diagram



프로그램 일부

```
import math

class Star:
    def __init__(self,name,mass,temperature):
        self.name=name
        self.mass=mass
        self.temperature=temperature
    def prns(self):
        print('name : {}, mass : {}, temperature :
{}'.format(self.name,self.mass,self.temperature))

class Main_sequence(Star):
    def __init__(self,name,mass,temperature):
        Star.__init__(self,name,mass,temperature)
        if temperature<4000:
            self.spectrum='M'
        elif temperature<5000:
            self.spectrum='K'
        elif temperature<6000:
            self.spectrum='G'
        elif temperature<7000:
            self.spectrum='F'
        elif temperature<10000:
            self.spectrum='A'
        elif temperature<25000:
            self.spectrum='B'
```

<그림 3> '객체지향 프로그래밍' 영역 프로젝트 과제 예

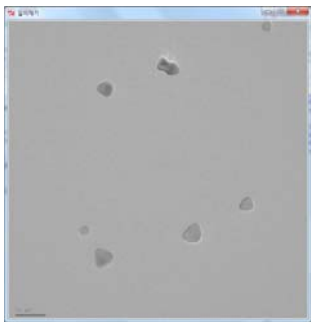
4.2.2 '객체지향 프로그래밍' 영역 수행 과제 및 평가

'객체지향 프로그래밍' 영역은 직접 클래스를 정의해 객체를 생성하여 프로그램을 완성하는 프로젝트 과제를 수행평가로 실시하였다. 이 과제를 수행하기에 앞서 다양한 실습 문제를 통해 어떻게 파이썬을 이용해 클래스를 정의하고 객체를 생성해야 되는지를 이해하고, 객체지향 프로그래밍이 구조적 프로그래밍의 이해와 어떠한 차이를 가지고 있는지를 이해하도록 하였다. <그림 3>은 학생이 수행한 프로젝트 과제의 한 예로, 객체지향 설계를 통해 직접 객체지향 프로그래밍을 구현한 프로젝트 과제 중 한 예이다.

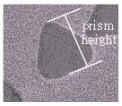
4.2.3 '시뮬레이션 프로그래밍' 영역 수행 과제 및 평가

'시뮬레이션 프로그래밍' 영역은 파이썬의 tkinter 모듈과 VPython을 이용한 GUI 프로그래밍 실습 후, 학생들은 자신의 관련 분야 과제를 수행하는데 필요한 시뮬레이션 프로그램을 개발하는 과제를 수행하였다. <그림 4>는 은 나노프리즘의 형성에 있어서 계면활성제의 분자량과 은 화합물 내의 음이온이 미치는 영향을 연구하면서 요구되는 은 나노 프리즘 높이를 재는 프로그램을 학생이 직접 작성한 예이다. 특히, 이 프로그램을 개발하게 되면서 학생은 은 나노 프리즘 높이를 재기 위해서는 비싼 프로그램을 사용해야 되는 한계를 극복할 수 있다는 설문 후기를 남겼다.

은 나노 프리즘의 높이 재는 실행 예	프로그램



```
>>> 32.202484376209235
39.44616584663204
33.54101966249684
```

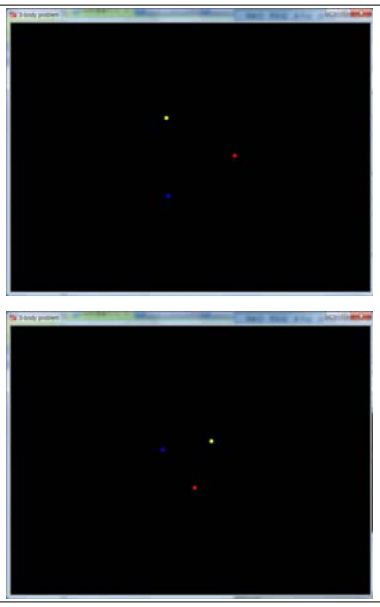
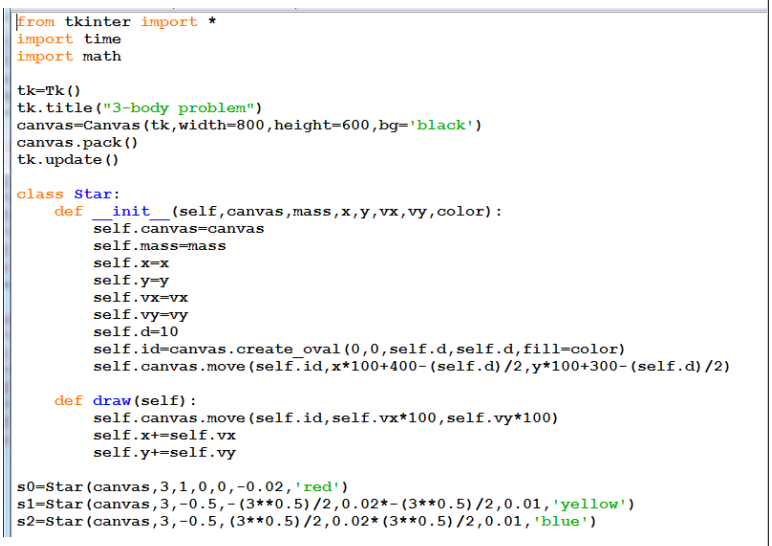



```
from tkinter import *
from math import sqrt
tk=Tk()
tk.title("길이재기")
pos=["",""]
i=0
canvas=Canvas(tk,width=650,height=650)
img=PhotoImage(file="c:\\A 10000 01 x40k.GIF")
canvas.create_image(0,0,anchor=NW,image=img)
label=Label(tk,text="")
canvas.pack()
tk.update()

def click1(event):
    global i
    if i==0:
        pos[0]=event.x
        pos[1]=event.y
        i=1
    elif i==1:
        print(sqrt((event.x-pos[0])**2+(event.y-pos[1])**2))
        i=0

canvas.bind all("<Button-1>",click1)
```

<그림 4> '시뮬레이션 프로그래밍' 영역 수행 예1 : 은 나노 프리즘 TEM 사진(40000배)과 높이 계산

<그림 5>는 학생이 만든 '3-body problem'으로, 3개의 천체를 우주 공간에 각각 질량, 속도를 부여하였을 때 나타나는 현상을 볼 수 있는 중력 시뮬레이션 프로그램 코드 일부분과 실행 예이다. 이 프로그램은 각 천체의 질량 초기 위치, 초기 속도를 숫자로 입력하면, 만유인력의 법칙에 따라 천체들이 운동하는 모습을 직접 볼 수 있게 설계되어 있다.

3체 문제는 수식으로는 답을 구할 수 없는 문제이기 때문에 컴퓨터 프로그램을 사용하여 근사적으로 운동을 분석할 수 있는 시뮬레이션 프로그램으로 적합한 내용이다.

3-body problem 실행 예	프로그램 일부
	

```
from tkinter import *
import time
import math

tk=Tk()
tk.title("3-body problem")
canvas=Canvas(tk,width=800,height=600,bg='black')
canvas.pack()
tk.update()

class Star:
    def __init__(self,canvas,mass,x,y,vx,vy,color):
        self.canvas=canvas
        self.mass=mass
        self.x=x
        self.y=y
        self.vx=vx
        self.vy=vy
        self.d=10
        self.id=canvas.create_oval(0,0,self.d,self.d,fill=color)
        self.canvas.move(self.id,x*100+400-(self.d)/2,y*100+300-(self.d)/2)

    def draw(self):
        self.canvas.move(self.id,self.vx*100,self.vy*100)
        self.x+=self.vx
        self.y+=self.vy

s0=Star(canvas,3,1,0,0,-0.02,'red')
s1=Star(canvas,3,-0.5,-(3**0.5)/2,0.02*-(3**0.5)/2,0.01,'yellow')
s2=Star(canvas,3,-0.5,(3**0.5)/2,0.02*(3**0.5)/2,0.01,'blue')
```

<그림 5> '시뮬레이션 프로그래밍' 영역 수행 예2 : 3-body problem 프로그램 개발

4.3 자기 설문 결과

학생들의 프로그래밍에 대한 인식을 확인하고자, 수업을 듣기 학기 초, 학기 후의 자기 평가 설문을 실시하였고, 평균 결과값은 다음과 같다 (Q# 수업 전/수업 후).

먼저, 학기 초에는 프로그래밍이 본인에게 공부하거나 연구하는데 도움이 된다고 생각하는 학생들이 많지 않았으나, 학기 후에는 프로그래밍이 본인에게 도움이 되고 계속 배우고 싶다는 생각을 많이 하는 것으로 나타났다 (Q1 2.96/4.64, Q3 2.4/4.48, Q4 3.2/4.36). 또한, 수업 전에는 학생들이 자신의 공부나 연구 분야에서 프로그래밍을 사용하는 부분에 대해 자신감이 많지 않았으나, 수업 후에는 프로그래밍에 대한 자신감이 향상된 학생들이 많았다 (Q2 2.68/4.08).

다음은 학생들이 프로그래밍 태도에 대한 주관식 설문의 일부분이다.

A : 객체지향 수업을 들으며 프로그래밍에 관심이 많이 생겨서 스스로가 기특하다고 생각했습니다.

B : 1학년 때부터 정보 과목 때문에 솔직히 많이 힘들었는데요. 이번 학기 동안 객체지향 수업 들으면서 실력도 늘은 것 같고 굉장히 즐거웠습니다.

두 번째로, 파이썬을 이용한 문제해결능력에 대한 자기평가 결과는 다음과 같다. 수업을 시작하기 전에는 파이썬(VPython 포함)을 해 본 학생은 한 명도 없었으며, 1명만이 클래스와 객체의 개념을 알고 있었다. 그러나 학생들은 수업 후 클래스를 정의하고 객체를 생성하는 프로그램을 작성할 수 있다는 설문결과를 보여주어 수업 전, 후가 큰 차이를 보여주었다(Q5 1.08/4.56, Q6 1.0/4.52). 그러나 수치 해석을 포함한 주어진 문제에 대한 해결전략을 구상해 알고리즘을 설계해 구현하는 능력은 수업 전, 후가 큰 차이를 보여주지 않았다 (Q7 3.76/4.52, Q8 3.88/4.36). 이 결과는 학생들이 1학년 때 C 프로그래밍 수업을 받은 경험이 크게 작용하였다고 볼 수 있다. 단, 학생들은 프로젝트 과제를 할 때 스스로 아이디어를 생성해 문제를 정의하는 것이 선생님이 제시해 준 문제를 해결하는 것보다 더 어려웠다는 주관식 설문 결과들이 있었다. 다음은 학생들이 문제를 해결하거나 정의하는 것에 대한 주관식 설문의 일부분이다.

C : 저한테 아주 잘 맞는 수업이었다고 생각합니다. 여러 가지 과제들을 할 때, 아이디어가

떠오르지 않았을 때는 긴장했지만 그래도 항상 과제를 해결하고는 뿌듯했습니다! 수학이 아니라 정보를 하고 싶다는 생각도 많이 했습니다!

D : 과제 등에서 아이디어를 내기가 엄청 힘들었습니다.:

마지막으로, 시뮬레이션 프로그램 개발에 관한 자기평가 결과는 다음과 같다. 학생들은 1학년 때에는 GUI 프로그래밍을 경험하지 못했기 때문에, 개인적으로 GUI 프로그래밍 공부를 한 극소수의 학생들을 제외하고 시뮬레이션 프로그램을 개발할 수 있는 학생들은 거의 없었다. 그러나 시뮬레이션 프로그래밍 수업과 프로젝트 과제를 통해, 어느 정도 자신의 공부나 연구 분야에서 필요한 시뮬레이션 프로그램을 개발할 수 있는 능력을 갖추게 되었다(Q9 1.12/4.04). 다음은 GUI 프로그래밍 수업이 자신에게 어떤 도움이 되었는지에 대한 주관식 설문 결과이다.

E : 너무 유익했어요. 파이썬으로 졸업논문 데이터 처리하는 게 더 편했어요.

F : 유명 GUI 프로그램들이 어떻게 만들어졌는지 생각을 해 볼 수 있게 되었습니다.

G : 천문 연구를 하는 데에는 꼭 필요한 수업인 거 같습니다.

설문결과, 전반적으로 학생들은 수업 전보다 프로그래밍 교육에 대한 중요성을 인식하고 있었고, 프로그래밍이 문제를 해결하거나 타 교과와 강한 연관성을 가지고 있다는 인식을 하는 것을 알 수 있었다.

5. 결론

본 연구는 프로그래밍 교육과정의 내용요소를 정보과학영재가 아닌 과학영재 고등학생들을 위한 도구 교과로서의 관점으로 설계하고 한 학기 동안의 수업을 통해 설계한 내용요소에 대해 검증하였다. 먼저, 1년 동안의 수업 결과를 바탕으로, 도구교과로서의 프로그래밍 교육과정의 내용요소를 프로그래밍의 이해, 객체지향 프로그래밍, 시뮬레이션 프로그래밍 영역으로 구성하여 설계하였다. 그 후, 설계된 내용요소를 바탕으로 한 학기동안 Python(RUR-PLE, VPython)을 이용하여 수업을 진행하면서 프로젝트 과제와 문제해결 평가를 수행하였다. 그 결과 다음과 같은 결과를 도출할 수 있었다.

먼저, 학생들이 수업전보다 수학, 과학 공부나 연구를 하는데 프로그래밍이 필요하다는 인식을 많이 갖게 되었다는 것을 알게 되었다. 또한, 프로그래밍을 통해서 문제를 해결하거나 시뮬레이션 프로그램을 만드는데 자신감을 갖게 된 학생들이 많게 되었다.

두 번째로, 학생들은 프로그래밍 수업이 단순히 프로그래밍 언어를 배우는 수업이 아니라, 문제에 대한 해결방법을 찾아 설계하는 것이 더 중요하다는 생각을 하게 되었다. 연구 결과는 학생들에게 수업을 통해 프로그래밍 교육이 단순히 코딩 교육이 아닌 정보과학적 사고(Computational Thinking)를 향상시키는 교육이라는 관점을 갖게 해 주었다.

마지막으로, 대부분의 학생들은 자신의 연구나 공부를 할 때 어떻게 프로그램을 만들어 사용해야 될지에 대해 이해하고 있었다. 설계한 내용요소를 바탕으로 한 수업은 정보 뿐 아니라 수학, 과학 공부나 연구에서 데이터 분석을 위한 수치해석 및 시뮬레이션 프로그래밍 역량을 향상 시킬 수 있음을 알 수 있었다.

본 연구가 미래의 자연과학, 공학자가 될 과학영재 학생들이 이론의 검증 및 실험을 수행할 수 없는 극한 상황이나 나노 또는 빅 데이터 처리 등의 융합 역량을 갖출 수 있는 정보교과의 가이드라인을 제시할 수 있을 것이라 기대한다.

참 고 문 헌

[1] 교육인적자원부(2007). **중학교 재량활동 교육과정 - 정보 -**. 교육인적자원부 고시 제2007-79호.

[2] 교육인적자원부(2007). **실과(기술・가정) 교육과정 - 정보 -**. 교육인적자원부 고시 제2007-79호.

[3] 교육과학기술부(2011). **중학교 선택 교과 교육과정 - 정보 -**. 교육과학기술부 고시 제2011-361호.

[4] 교육과학기술부(2011). **실과(기술·가정) 교육과정 - 정보 -**. 교육과학기술부 고시 제 2011-361호.

[5] 김종혜(2009). **정보과학적 사고 기반의 문제해결 능력 향상을 위한 중등 교육 프로그램**. 고려대학교대학원 박사학위논문.

[6] Committee on Education and the Workforce(2011). H.R. 3014 (112th) : Computer Science Education Act of 2011. *Retrieved from https://www.govtrack.us/congress/bills/112/hr3014/text.*

[7] Mashable(2013). Finland Eyes Programming Classes for Elementary School Students. *Retrieved from http://mashable.com/2013/11/16/finland-tech-education-schools/.*

[8] Teaching the new computing curriculum(2013). *Retrieved from https://www.gov.uk/government/news/teaching-the-new-computing-curriculum.*

[9] 미래창조과학부(2013). 과학영재 발굴·육성 종합계획(안) ('13~'17). *Retrieved from http://www.nstc.go.kr/.*

[10] IASA(Israel Arts and Sciences Academy). *Retrieved from http://iasa.excellence.org.il/english.html.*

[11] NUS High School of Math & Science. *Retrieved from http://www.nushigh.edu.sg/.*

[12] The Nobel Prize in Chemistry 2013(2013). *Retrieved from http://www.nobelprize.org/nobel_prizes/chemistry/laureates/2013/.*

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

[14] 김자미, 이원규(2010). 교과교육의 측면에서 본 정보교과의 정체성에 대한 고찰. **한국정보교육학회 논문지, 14(2)**, 219-227.

[15] 이영준, 이은경(2008). 정보교육의 본질과 전망. **한국컴퓨터교육학회논문지, 11(3)**, 1-11.

[16] Denning,P.(2007). Computing is a Natural Science. *Communications of the ACM, 50(7)*, 13-18.

[17] 정여진(2010). **기초도구교과로서의 정립을 위한 중·고등학교 정보 교육과정과 대학 교육과정의 연계성 연구**. 고려대학교.

[18] Sheikh Iqbal Ahamed, Dennis Brylow,Rong Ge, Praveen Madiraju, Stephen J. Merrill, Craig A. Struble, James P. Early(2010). Computational Thinking for the Sciences. *Proceeding of SIGCSE'10*, 42-46.

[19] Vicki Allan, Valerie Barr, Dennis Brylow, Susanne Hambrusch(2010). Computational Thinking in High School Courses. *Proceeding of SIGCSE'10*, 390-391.

[20] Owen Astrachan, Susanne Hambrusch, Joan Peckham, Amber Settle(2009). The Present and Future of Computational Thinking. *Proceeding of SIGCSE'09*, 549-550.

[21] Valerie Barr, Chris Stephenson(2011). Bringing Computational Thinking to K-12 : What is Involved and What is the Role of the Computer Science Education Community? *ACM Inroads, 2(1)*, 48-54.

[22] Susanne Hambrusch, Christoph Hoffmann, John T. Korb, Mark Haugan, Antony L. Hosking(2009). A Multidisciplinary Approach Towards Computational Thinking for Science Majors. *Proceeding of SIGCSE'09*, 183-187.

[23] TIOBE programming community index (2013). TIOBE index. *Retrieved from http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.*

김 종 혜

2005 고려대학교 컴퓨터교육과(교육학석사)
2009 고려대학교 컴퓨터교육과(이학박사)
2010~현재 경기과학고등학교 교사
관심분야: 정보 교육과정, 정보영재
E-Mail: jonghye.kim@inc.korea.ac.kr