https://doi.org/10.32431/kace.2019.22.2.004

텍스트 기반 프로그램 평가에서 부분점수 구성에 관한 고찰

이재영† · 김자미†† · 이원규†††

요 약

SW 개발과 관련된 프로그램 평가는 학생이 작성한 결과물에 대한 정답 여부만을 제공하는 경우가 많다. 본 연구는 프로그램 평가의 내용이 교사의 수업을 지원하고, 부분점수 제공에서 어떤 부분을 중요하게 고려해야 하는지에 대한 기초자료를 제공하기 위한 목적으로 진행되었다. 목적 달성을 위해 본 연구는 자유학기제 운영 중학교 90명을 대상으로 2개월 간 파이썬 수업을 진행하고, 과정에서 수집된 1185개의 소스코드를 분석하였다. 분석 결과, 학생의 실수가 많은 오류는 '문법오류'이며, 교사들은 '논리오류'를 중요하게 고려하였다. 결과를 토대로 학생의 잦은 문법오류를 줄일 필요가 있으며, 교사는 평가에서 논리적 측면에 대한 중요성을 고려하여 학생들의 프로그램을 평가하고, 부분점수를 고안할 필요가 있다. 본 연구는 프로그램 평가를 학습 지원과 평가의 관점으로 고려했다는 점에 의의가 있다.

주제어 : 프로그램 평가, 교육 평가, SW 교육

A Study on Partial Scoring in Text Based Program Evaluation

JaeYoung Leet · JaMee Kimtt · WonGyu Leettt

ABSTRACT

The evaluation of programs related to SW development often only provides the right answer of the student's program. The purpose of this study was to provide the baseline data about the contents of the program evaluation support the teacher's class and which part should be considered important in partial scoring. To accomplish the goal, we had two months of Python lessons for 90 middle school students in free-semester and analyzed 1185 source codes collected during the lessons. Result of analysis, many students made mistakes about syntax errors and teachers considered logic errors as important. Based on the result, it is necessary to reduce the student's syntax errors and teachers need to evaluate student's program with considering the importance of logical aspects and necessary to devise a partial scoring. This study has significance about consideration of program evaluation from the perspective of learning support and evaluation.

Keywords : Program Evaluation, Education Evaluation, SW Education

†정 회 원: 고려대학교 일반대학원 컴퓨터학과 석사
††종신회원: 고려대학교 교육대학원 컴퓨터교육전공 조교수
†††종신회원: 고려대학교 정보대학 컴퓨터학과 교수 (교신저자)
논문접수: 2019년 1월 31일, 심사완료: 2019년 3월 15일, 게재확정: 2019년 3월 18일
* 본 논문은 2016년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2016RIA2B4014471)

1. 서론

현대 사회는 정보통신기술이 기존 산업·서비스·신기술 등과 결합되어 경제·사회 전반에 걸쳐 사회·산업 구조의 변화를 이끌고 있다. 기술의 변화는 사회 뿐 아니라 교육의 변화로 이어졌다, 세계 각국은 교육과정 개정을 통해 컴퓨팅 사고력의 향상에 주력하고 있다[1].

영국은 2014년 컴퓨팅(Computing)을 개설하여 전 학년 필수과목으로 지정하였고, 인도는 초등학교부터 컴퓨터 과학 중심의 정보교육을 필수화하였다. 미국의 CSTA(Computer Science Teachers Association)는 2016년 9월 K-12 Computer Science Framework를 발표하였고, 이스라엘은 중등교육에서 대학 수준의 컴퓨터과학을 다루고 있다[2][3][4].

한국에서는 2015 개정 교육과정을 통해 정보 교육을 강화하였고, 중등학교에서는 블록 기반, 고등학교에서는 텍스트 기반 프로그래밍 언어를 학습을 권고하였다[5]. 한국과 달리 미국, 인도에서는 4학년에는 블록 기반 프로그래밍 언어를, 6학년부터는 텍스트 기반 프로그래밍 언어를 학습하도록 권장하고 있다[6][7].

프로그래밍 교육 강화로 인해 학생의 프로그램 관련 역량에 대한 평가 역시 중요해지고 있다. 프로그래밍 교육에서 평가는 학생이 작성한 결과물인 프로그램을 기반으로 이루어지며, 프로그래밍 교육 목적 달성 여부를 정확하게 판단할 수 있어야 한다[8][9]. 교육에서 평가를 교육 목적 달성 여부를 파악하고, 학생의 학업 성취 수준을 확인할 수 있도록 구성되어야 한다[4]. 평가의 내용 뿐 아니라 학생의 성취를 정확히 평가하기 위한 방법 또한 중요하게 고려되어야 한다.

중등교육의 정보과에서는 프로그램 평가 방법으로 프로그래밍의 기본 개념과 원리, 프로그램의 정확성과 효율성, 문제 해결 목적에 적합한 변수, 자료형, 연산자, 입력과 출력, 제어 구조, 배열, 함수의 사용 등을 제시하고 있다[5]. 다양한 문헌들에서는 프로그램 평가에 대한 논의들이 이루어지고 있으나, 2015 개정 교육과정에 근거하여 평가 방법을 논의한 학술적 연구들은 미비한 편이다.

프로그램을 평가하기 위해 고등이나 평생교육 관점에서는 Online Judge와 같은 온라인 평가 시스템을 활용하고 있으며, 초중등 교육에서는 프로그램을 보다 구체적으로 분석하기 위한 평가 루브릭 등이 활용되고 있다. 그러나 온라인 평가 시스템은 정답과 오답에 대한 측면만을 제시하는 경우가 많다. 평가 루브릭은 이전의 프로그램 분석을 통한 것이 아니라 평가 요소를 추출하고, 그에 따른 배점 제공의 형태이다. 즉, 학교 현장에서 교사들이 학생들에게 프로그래밍을 가르칠 때, 학습 지원에 도움을 주기 위한 측면이라거나, 평가 피드백을 통해 학습자 스스로 무엇을 틀리고 맞았는지를 확인할 수 있도록 부분 점수를 구성하지는 않았다.

이에 본 연구는 초등학교부터 진행되어야 하는 교육의 관점에서 프로그램 평가를 논의하기 위해 선행 연구에 대한 검토를 통해 현재의 상태를 확인하였다[5][10][11][12]. 그리고 학교 현장에서 교사들이 수업을 진행하는 데 있어서 학생의 진행 내용을 고려할 수 있도록 하는 근거를 마련하기 위한 목적으로 진행되었다.

교사들의 프로그램 수업 지원의 관점에서 평가가 이루어질 수 있도록 고려한 본 연구는 첫째, 프로그래밍 언어를 학습하는 과정에서 다양한 오류의 종류와 유형에 대한 관련 연구 분석을 통해 본 연구의 활용 언어인 파이썬으로 오류 유형을 정리하였다. 둘째, 프로그램 평가가 교사들의 수업 지원에 도움을 줄 수 있도록 학생이 오류를 자주 발생시키는 부분을 검토하고, 교사들이 평가에서 중요하게 고려하고 있는 부분을 파악하여 정오 형태가 아닌 부분 점수를 줄 수 있도록 하는 기초자료를 제공하고자 한다.

2. 관련 연구

2.1 온라인 평가 시스템

온라인 평가 시스템은 프로그램 관련 문제 풀이에 대해 사용자에게 실시간으로 평가 결과와 피드백을 제공한다[13]. 온라인 평가 시스템에서는 프로그램의 정확성과 효율성을 평가한다. 프로그램 정확성은 문제에 대한 입출력 형식을 제한하고, 미리 준비된 입력 값에 대한 출력 값 셋을 사용자가 작성한 프로그램 출력 결과와 비교하여 평가한다.

프로그램 효율성은 해당 문제에 대한 프로그램 실행 시간 제한, 최대 메모리 사용량 제한 등을 두어 공간 복잡도를 평가한다.

온라인 평가 시스템을 제안한 Kurnia (2001)의 연구는 모델을 설계하고, 고려해야 할 사항과 필요한 요소 등을 제안하였다[13]. 온라인 평가 시스템을 활용한 프로그래밍 언어 학습 효과 분석에는 Brito (2014)와 Wang (2016)의 연구가 있다.

주 단위의 자가 평가 활동과 실시간 평가 내용에 대한 피드백을 제공한 Brito (2014)는 온라인 평가 시스템이 프로그래밍 학습을 돕기 위한 좋은 방법이라 제시하였다[14]. Wang (2016)은 온라인 평가 시스템을 활용한 대학생의 프로그래밍 언어 학습이 더 높은 효율을 보이며, 지속적으로 학생의 흥미와 관심을 자극하여 학습을 지원할 수 있는 도구라고 하였다[15]. 즉, 온라인 평가 시스템은 실시간 평가 결과 및 피드백 제공으로 높은 프로그래밍 언어 학습 효율을 얻을 수 있다. 또한 학생이 어려워하거나 이해가 부족한 개념에 대해 시기적절한 대처가 가능함을 알 수 있다[13][15].

중등교육에서 시행되는 프로그래밍 교육은 학생의 문제해결력을 향상시키는데 목적이 있다[5]. 중등학생에게 온라인 평가시스템을 활용하거나, 일반 서술형의 평가에서도 프로그램에 대해 정답 여부가 아니라 학습자의 수준에 맞는 피드백을 제공할 필요가 있다. 즉, 프로그램 소스코드를 분석하여 정답 여부의 구분 없이 어떤 측면에 대한 이해가 부족한 지를 파악할 수 있도록 할 필요가 있다.

2.2 프로그램 평가 루브릭

루브릭은 학생의 수행 평가 점수 산정을 제한하기 위해 명시적으로 기준을 제공하여 평가자가 피드백을 제공하는데 도움을 주며, 평가의 신뢰성을 향상시킬 수 있다[16]. 프로그래밍의 경우 주어진 문제를 해결하는데 다양한 방법이 존재하기 때문에 객관적인 평가를 위한 기준을 마련할 필요가 있다. 프로그래밍을 객관적으로 평가하기 위한 명시적인 기준으로 평가 루브릭(Scoring rubric)이 있다[12][16][17].

프로그램 평가 루브릭에 대한 선행연구는 다음과 같다.

첫째, 초중등학생을 대상으로 한 프로그램 평가 루브릭을 개발한 Brennan (2012)은 학업 성취 수준을 파악하기 위해 3개 영역과 6가지 요소로 구성된 CT Framework를 개발하였다[20]. 유사하게 안성훈(2016)도 학생의 프로그래밍을 다양한 관점에서 평가하기 위해 교육적 가치 측면과 SW 품질적 측면의 평가 기준을 제시하였다[11]. 안성훈은 교육적 측면의 가치를 고려한 대신 일반 SW 개발에서 사용하는 품질에 대한 측면을 포함하여, 초중등학생용으로는 활용하기 어려운 점이 있었다.

둘째, 대학생을 대상으로 한 김민자 (2017)는 비전공자 대상 컴퓨터과학 강의에서 학생의 산출물을 CT관점으로 평가하기 위한 루브릭을 고안하였다. 평가 내용은 3개 영역에 대한 10가지 역량요소와 평가 요소로 구성되었으며, 각 영역의 역량요소에 대한 평가 요소를 기반으로 학생의 수준에 따라 1-3점, 4-6점, 7-10점을 산정하는 점수 체계를 포함하고 있다[17]. 비전공자를 대상으로 한 김재경 (2017)은 AP Computer Science Principles Curriculum Framework에서 제시한 Computational Thinking Practices 모델의 6가지 구성요소를 기반으로, 추상화 단계의 컴퓨팅 개념 지식과 구현 단계의 프로그래밍 실습 평가 등 2개의 루브릭을 고안하였다. 추상화 단계와 구현 단계의 루브릭은 컴퓨팅 사고력의 세부 요소를 4개의 분류와 8가지 항목으로 구분하여 미숙함, 좋음, 뛰어남의 3단계로 구성되었다[12]. 그러나 김재경의 연구는 컴퓨팅 사고력을 임의로 4개로 분류하였다는 점에 대한 근거가 미약함을 알 수 있다.

예비교사를 대상으로 CT 역량에 대한 루브릭을 개발한 최형신 (2014)은 절차 및 알고리즘, 병행화 및 동기화, 자료 표현, 추상화, 문제 분해, 시뮬레이션 등 6가지 CT 세부역량을 제시하고, 기초, 발달, 능숙 3단계로 개발하였다[18]. 최형신의 경우, CSTA에서 제시한 9개의 CT 요소에 기반한 것으로 요소의 모호성, 병행화의 내용에 대한 기술 부족 등의 한계를 갖는다. 모바일 환경에서 대학생들의 CT를 평가 루브릭을 개발한 Sherman (2015)은 모바일 환경에 대한 CT(Mobile CT)를 제시하였다. Mobile CT 평가를 위한 4점으로 구분하였다[19].

이상의 논의들을 분석하면 다음과 같다.

첫째, 루브릭의 평가 요소는 정량적 측정이 가능한 형태로 구성할 필요가 있다. 즉, 평가 내용에 대한 모호한 기술이나, 평가 내용에 대한 모호성은 루브릭이 학교 현장에서 사용될 때 한계를 가질 수 있기 때문이다.

둘째, 프로그래밍은 많은 경험을 통해 학습하게 되는 능력으로 문제해결력, 논리적 사고력, 프로그래밍 문법 이해 등의 역량이 요구된다. 프로그램을 구성하는 과정에 대한 평가보다는 오답에 집중한 루브릭은 교사가 학생에게 어떤 피드백을 제공해야 하는지에 대한 통찰을 제시하기 어렵다. 학생 입장에서는 스스로 오류의 원인과 이유를 찾아 해결해 가는 과정에서 어려움을 느낄 수 있으며, 학업 포기로 이어질 수 있음을 고려해야 할 것이다.

2.3 프로그램 오류 유형

프로그램 오류는 프로그래밍 언어의 문법에 맞지 않거나, 논리 구조가 맞지 않게 작성되었을 때 발생하는 것을 말한다. 프로그램 오류 유형에 대한 선행연구를 정리한 것은 <표 1>과 같다.

<표 1>의 내용을 분석하면, 첫째, 특정 프로그래밍 언어의 특징이 반영되어 오류 유형이 다르게 나타난 경우로, 문외식 (2006)의 연구에서 분류한 인터페이스 오류, 남재원 (2011)의 연구에서 분류한 로봇 제어 오류 등이 있다[21][22].

둘째, 오류 유형에 대한 오류 종류가 다르게 분류되어 나타난 경우로, Murray (1987)의 연구에서 문법오류를 통사적 오류, 코딩 오류로 분류하였고, 논리오류를 잘못된 문제 해석의 오류, 전략적 오류로 분류한 것이 있다[25].

이상을 토대로 할 때, 오류 유형을 문법오류, 의미오류, 논리오류로 재정리할 수 있다.

<표 1> 프로그램 오류유형 선행연구

연구자	연구대상	언어	오류 유형
문외식	초등학생	비주얼베이직6	논리 오류
			데이터 취급 오류
			데이터 정의 오류
			인터페이스 오류
			설계 오류
			연산 오류
			기타 오류
장혜선	초등학생	두리틀	어휘 오류
			구문 오류
			의미 오류
			논리 오류
남재원	초등학생	NXC 로봇 프로그래밍	문법 오류
			논리 오류
			코딩 오류
			로봇 제어 오류
김지선	중등학생	C언어	문법 오류
			논리 오류
			코딩 오류
Murray	대학생	C언어	잘못된 문제 해석
			전략적 오류
			번역상의 오류
			통사적 오류
			코딩 오류
Vipindeep	대학생	C / C++	포인터와 메모리
			Alias 오류
			동기화 오류
			데이터 산술연산
			연산자 중복 오류
			클래스 관련 오류

다양한 문헌 검토와 파이썬의 프로그램 분석 등의 과정을 통해 본 연구는 <표 2>와 같이 파이썬에 근거하여 오류 유형과 오류 종류를 정리하였다[21][22][23][24][25][26].

<표 2> 파이썬 오류 유형과 오류 종류

오류 유형	오류 종류	오류 정의
문법오류	실수로 인한 오탈자	• 잘못된 키워드 입력 • 쌍따옴표(“ ”) 등의 누락 또는 잘못된 사용
	들여쓰기 오류	• 문법에 어긋난 들여쓰기 사용
	기본 문법 이해	• 문법 이해 부족으로 조건문, 반복문, 함수 등의 잘못된 사용
정적 의미오류	선언하지 않은 변수 사용	• 변수 선언을 누락하고 변수 사용
	부적절한 연산 시도	• 부적절한 등식(=), 부적합한 피연산자 사용과 같은 문법에 적합하지 않은 연산 시도
동적 의미오류	0으로 나누기	• 0으로 나누기 연산 시도
	부적절한 인덱스 사용	• 참조 불가능한 변수를 참조를 위한 인덱스에 사용

동적 의미오류	인덱스 범위 오류	• 변수가 갖는 값의 범위를 벗어난 인덱스 사용
	부적절한 매개변수 오류	• 함수 호출에 필요한 매개변수 누락 • 부적합한 매개변수 사용
	논리식 또는 조건식 누락	• 제어구조에 필요한 논리식 또는 조건식의 누락
논리오류	불필요한 명령어 포함	• 문제해결에 필요하지 않은 명령어가 포함되어 실행결과가 달라지는 경우
	필요한 명령어 누락	• 문제해결에 필요한 명령어를 작성하지 않아 올바른 실행 결과가 나오지 않는 경우
	출력 누락	• 프로그램 실행 결과가 모범답안과 같으나 프로그램 실행 결과를 출력하지 않은 경우
	부적절한 값 설정	• 잘못된 값을 설정하여 올바른 실행결과가 나오지 않는 경우
	부적절한 조건 설정	• 논리식/조건식이 문제해결에 맞지 않게 설정된 경우 • 문제해결을 위해 필요한 반복 횟수와는 다른 반복 횟수가 설정된 경우(무한루프)
	잘못된 명령어 순서	• 작성한 명령어의 순서 또는 위치가 잘못된 경우

<표 2>와 같이 파이썬의 오류는 3가지로 구분하여 분석하였다. 분석 결과, 첫째, 문법오류는 파이썬 언어의 문법을 위배했을 때 발생하는 오류로 컴파일이 정지되며, 프로그램 실행이 되지 않는다. 문법오류에는 3가지 오류 종류가 포함된다. 실수로 인한 오탈자는 부주의로 인해 키워드를 잘못 입력한 경우 또는 print()문이나 문자열 등에서 쉼표(,), 따옴표(‘ ’), 쌍따옴표(“ ”)가 누락되거나 잘못 사용된 경우에 발생하는 오류이다. 기본 문법 이해는 파이썬 문법 이해 부족으로 조건문, 반복문, 함수 등의 사용에서 발생하는 오류이다. 들여쓰기 오류는 C언어의 코드 블록을 잘못 설정한 경우 등을 고려하였다.

둘째, 의미오류는 작성한 소스코드의 해석과 관련된 것으로 정적의미오류(2가지)와 동적의미오류(5가지)로 분류하였다. 정적의미오류는 프로그램 실행 전에 확인할 수 있으며, 변수 선언, 연산자 사용에 대한 오류이다. 동적의미오류는 프로그램 실행 과정에서 동적으로 발생하는 오류로 0으로 나누기, 부적절한 참조에 의해 발생하는 오류로 참조를 위한 인덱스, 함수 호출에 사용되는 매개변수 사용 등을 포함하였다.

셋째, 논리오류는 작성된 프로그램에 문법오류나 의미오류는 없으나, 프로그램 실행 결과가 문제에서 요구한 내용과 다른 경우에 발생한다. 문제해결에 필요하지 않은 명령어가 포함되거나, 명령어가 누락된 경우, 명령어의 순서, 반복 횟수나 변수 잘못 사용 등에 대한 것으로 규정하였다.

3. 연구방법

3.1 수업 및 문제구성

프로그램 평가를 통해 교사의 수업을 지원하고, 평가의 명확성을 고찰하기 위한 본 연구의 대상은 자유학기제 수업에 참여한 서울의 A 중학교의 90명 학생들이다. 학생들은 프로그래밍 경험이 없는 중학교 1-2학년 이다.

수업의 구성은 다음과 같다.

언어는 파이썬으로 현재 미국, 영국 등의 중학생 교육에 활용되고 있으며, 범용적으로 활용이 많은 오픈소스를 활용하였다.

수업은 매주 2시간 8주간 진행되었다.

수업 내용은 프로그래밍과 프로그래밍 언어의 개념, 파이썬 문법, 입력과 출력, 변수와 연산자, 조건문, 반복문, 중첩 반복문, 리스트와 배열로 구성하였다.

3.2 프로그램 분석의 절차 및 방법

수업을 통해 수집된 전체 1185개의 프로그램에 대해 다음의 내용에 따라 학생의 프로그램을 평가하고 발생한 오류를 분석하였다.

첫째, 제시한 문제에 대해 최종적으로 제출한 프로그램을 평가하였다.

둘째, 오류 메시지에 근거하여 오류 종류와 발생원인을 분석하였다.

셋째, 처음 오류가 발생한 지점을 기준으로 작성된 내용을 평가에 반영하였다. 반면에, 오류 분석은 프로그램에서 발생한 전체 오류를 포함하여 분석하였다.

넷째, 다양한 오류를 파악하기 위해, 입력과 출력, 변수 및 자료형, 연산자, 제어구조(조건문과 반복문)의 개념을 포함하고 있는 중첩 반복구조 유형의 문제에 대한 풀이 내용을 분석하였다.

전체 1185개 프로그램 중 중첩 반복구조 문제에 대한 프로그램은 118개였다. 중첩 반복 구조 유형의 문제를 예로 설명하면 <표 3>과 같다.

<표 3> 중첩 반복구조 문제 예시

문제	모범 답안	출력 예시

중첩 반복문을 사용하여 2~9단까지 구구단을 출력하는 프로그램을 작성하시오.	for a in range(2, 10):
	  for b in range(1,10):
	  print(a,"*",b,"=",a*b)
		2 * 1 = 2
		2 * 2 = 4
		2 * 3 = 6
		(중략)
		9 * 7 = 63
		9 * 8 = 72
		9 * 9 = 81

각 문제에 대해 최소 3개 이상의 모범 답안을 작성하여, 평가를 하였다. 만약 오류가 있을 경우, 문법오류와 정적의미오류는 컴파일러가 제공하는 오류 메시지의 내용에 근거하여 분석하였다. 동적의미오류는 프로그램 실행 중에 발생한 오류 내용에 따라 분석하였다. 논리오류는 모범 답안과 비교하여 올바르지 않게 작성된 내용을 파악하고, 오류 원인을 분석하였다.

3.3 중요한 오류 추출을 위한 설문

교사를 대상으로 실제 발생한 오류들 중 어떤 오류가 심각한 오류인지를 확인하는 설문을 진행하였다. 설문의 구성 및 절차는 다음과 같다.

첫째, 문제 유형에 따라 발생할 수 있는 오류의 중요성을 파악하기 위해, 분석한 오류 내용에 근거하여 Likert 5점 척도로 설문을 구성하였다.

둘째, 설문의 구성은 <표 4>와 같다.

<표 4> 설문 구성 예시

오류예시	발생한 오류	오류 발생 원인	매우 중요	중요	보통	중요하지 않음	전혀 중요하지 않음
a=6 b=3 print("a+b"=9)	실수로 인한 오탈자	print()함수 내 쌍따옴표(“) 위치가 적합하지 않음			V		

<표 4>와 같이 각 문제 유형에 대한 오류 예시, 오류 종류, 발생 원인을 설명하고, 해당 문제 유형에서 각 오류 종류가 갖는 중요성을 묻는 형식으로 설문을 구성하였다.

셋째, 설문 대상으로 5년 이상의 교사 경력, 교과서 집필 경험, 석사 이상의 학위소지자, 텍스트 기반 프로그래밍 수업 경험 3년 이상의 조건 중 2개 이상 부합하는 중학교 교사와 고등학교 교사 13명을 선정하였다.

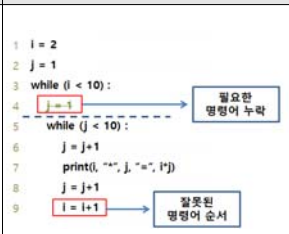
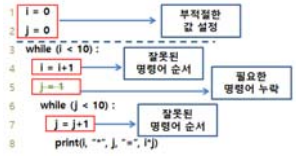
4. 연구결과

4.1 프로그램 오류 사례

문법오류는 컴파일 과정에서 확인 할 수 있고, 의미오류는 프로그램 실행 중에 확인을 할 수 있기 때문에 오류가 발생한 내용에 대해 비교적 쉽게 파악할 수 있다. 이와 달리 논리오류는 프로그램 실행 결과를 통해 발생 여부를 확인할 수 있기 때문에, 소스코드를 분석하여 발생한 오류 내용에 대해 파악할 수 있다.

중첩 반복구조 문제 유형에서 학생이 실수한 오류 종류와 오류가 발생한 원인을 정리한 결과는 <표 5>와 같다.

<표 5> 중첩 반복구조 문제 유형 오류 분석 내용

학생	소스코드	오류원인	
A		필요한 명령어 누락	반복 시작을 위한 값을 대입하는 명령어 누락
		잘못된 명령어 순서	변수 i에 대한 증감식의 잘못된 위치
B		부적절한 값 설정	변수 i, j에 잘못된 값을 설정
		잘못된 명령어 순서	변수 i에 대한 증감식의 잘못된 위치
		필요한 명령어 누락	반복 시작을 위한 값을 대입하는 명령어 누락
		잘못된 명령어 순서	변수 j에 대한 증감식의 잘못된 위치

A학생의 경우 inner while문에서 사용하는 변수 값을 변경하기 위한 명령어가 누락되어, 구구단이 2단만 출력되기 때문에 필요한 명령어 누락으로 분류하였다. 또한, outer while문의 증감식을

inner while문에 작성하여 올바른 구구단이 출력되지 않으므로 잘못된 명령어 순서로 분류했다.

B학생의 경우 변수 i와 j에 잘못된 값 대입으로 구구단이 1단부터 출력되기 때문에 논리오류의 부적절한 값 설정으로 분류했고, outer while문에서 값을 증가시킨 후 출력하여 잘못된 실행결과가 출력되므로 잘못된 명령어 순서로 분류했다. 또한, inner while문에서 매 반복의 시작 값을 대입하는 명령어가 누락되어 필요한 명령어 누락으로 분류했고, inner while문에서 값을 증가시킨 후 출력하여 올바른 실행결과가 출력되지 않기 때문에 잘못된 명령어 순서로 분류했다.

학생이 실수한 오류를 분석한 결과, <표 3>의 A학생과 B학생 모두 변수에 대한 증감식을 잘못된 위치에 작성하였지만, A학생과 B학생이 실수한 위치가 서로 다른 것과 같이 프로그래밍 개념 및 문법 이해 수준에 대한 개인차로 인해 같은 종류의 오류라 하더라도 오류가 발생한 원인은 다양하게 나타났다.

중첩 반복구조 문제 유형에서 나타난 오류 유형으로는 문법오류, 의미오류, 논리오류 모두 고르게 나타났으나 논리오류의 비중이 높았다. 이를 수업 과정에서 학생들에게 확인한 결과 반복 구조에 대한 이해는 비교적 쉽게 할 수 있었으나, 중첩 반복구조에 대한 개념을 이해하는 과정에서 많은 어려움을 겪는다는 것을 알 수 있었다. 또한, 같은 종류의 오류라고 하더라도 각 오류가 발생한 원인이 다르기 때문에, 이를 고려하여 학생이 이해를 하지 못한 부분에 대한 명확한 파악을 통해 학습을 지원하기 위한 시기적절한 대처가 필요하다.

4.2 문제 유형에 따른 오류의 중요성

문제 유형별로 발생한 오류 종류를 살펴보면, 변수와 연산자는 실수로 인한 오탈자, 기본 문법 이해(이상 문법오류), 선언하지 않은 변수 사용, 부적절한 연산시도(이상 정적의미오류), 불필요한 명령어 포함, 필요한 명령어 누락, 부적절한 값 설정, 출력 누락(이상 논리오류)등 8종의 오류가 나타났다. 조건구조, 반복구조, 중첩 반복구조는 변수와 연산자 유형에서 발생한 8종의 오류를 포함하여, 들여쓰기 오류(이상 문법오류), 논리식 또는 조건식 누락(이상 동적의미오류), 부적절한 조건 설정, 잘못된 명령어 순서(이상 논리오류)등 12종의 오류가 나타났다.

오류 유형별로 살펴보면 문법오류가 188회, 의미오류가 129회(정적의미오류 87회, 동적의미오류 42회), 논리오류가 103회 발생했다. 문법오류에서는 141회로 기본 문법 이해 오류가 가장 많이 나타났고, 다음으로 들여쓰기 오류가 32회 나타났다. 의미오류에서는 58회로 부적절한 연산시도가 가장 많이 나타났고, 다음으로 논리식 또는 조건식 누락이 42회 나타났다. 논리오류에서는 21회로 부적절한 값 설정이 가장 많이 나타났고, 다음으로 필요한 명령어 누락이 19회 나타났다.

문제 유형에 따른 오류의 중요성은 다음과 같다.

변수와 연산자 유형에서는 부적절한 연산 시도, 출력 누락, 필요한 명령어 누락 순으로 중요성이 높게 나타났다. 조건 구조 유형에서는 필요한 명령어 누락, 잘못된 명령어 순서, 논리식 또는 조건식 누락 순으로 나타났고, 반복 구조 유형에서는 논리식 또는 조건식 누락, 필요한 명령어 누락, 잘못된 명령어 순으로 나타났다. 중첩 반복구조 유형에서는 잘못된 명령어 순서, 필요한 명령어 누락, 부적절한 조건 설정 순으로 나타났다.

<표 6>은 중첩 반복구조 문제 유형에서 발생한 오류 종류에 대한 오류 횟수, 빈도, 중요성을 나타난 것이다.

<표 6> 중첩 반복구조 문제 유형에서 발생하는 오류 종류 및 중요성

오류유형	오류 종류	발생 횟수	빈도(%)	중요성 평균값
문법오류	실수로 인한 오탈자	4	4.3	2.53
	들여쓰기 오류	17	18.1	3.30
	기본 문법 이해	21	22.3	3.84
정적 의미오류	선언하지 않은 변수 사용	4	4.3	3.53
	부적절한 연산시도	8	8.5	3.63
동적 의미오류	논리식 또는 조건식 누락	7	7.4	4.07
논리오류	불필요한 명령어 포함	8	8.5	3.84
	필요한 명령어 포함	7	7.4	4.46
	부적절한 값 설정	2	2.1	3.84
	부적절한 조건 설정	5	5.3	4.41
	잘못된 명령어 순서	7	7.4	4.60
	출력 누락	4	4.3	3.63

중첩 반복구조 문제 유형에서 가장 많이 발생한 오류는 문법 오류의 기본 문법 이해, 들여쓰기 오

류였다. 오류 유형별로 보면 문법오류, 논리오류, 의미오류 순으로 나타났으나, 오류의 중요성은 논리 오류, 의미 오류, 문법 오류 순으로 나타났다.

오류가 발생한 횟수를 통해 학생들이 많이 실수하는 오류는 문법오류임을 알 수 있다. 이를 통해, 중학생은 프로그래밍 언어를 학습하는 과정에서 프로그래밍 개념과 문법을 이해하는 것에 많은 어려움을 겪는다는 것을 알 수 있다.

반면에, 설문 결과를 보면 논리 오류가 문법오류보다 높게 나타났다. 즉, 프로그램 평가를 함에 있어 프로그래밍 교육 목적에 따라 문제해결력, 논리적 사고력을 평가할 수 있는 논리오류를 가장 중요하게 고려하는 것을 알 수 있다.

4.3 부분점수 구성에 대한 고찰

프로그램 오류 분석과 설문을 통해, 학생들이 자주 실수하는 오류와 교사가 평가를 함에 있어 중요하게 생각하는 오류가 다름을 확인하였다.

프로그래밍은 많은 경험을 통해 학습하게 되는 능력으로, 학습 초기 단계에서 프로그래밍 개념과 문법을 이해하지 못할 경우, 그 후 단계를 학습하기에 어려움이 따른다. 이를 극복하기 위해, 학생들이 체험을 통해 학습해가도록 지도해야 한다.

또한, 학생이 무엇을 이해하였고, 어떤 부분에서 어려움을 겪는지에 대한 측면을 파악하여, 그에 맞는 적절한 피드백을 제공하여 학습을 지원하는 것이 필요하다.

부분점수 구성을 통해, 학습의 측면에서 학습자 전반의 상황을 보다 명확하고 구체적으로 파악해서 학습을 지원하고, 보다 객관적인 평가가 진행될 수 있다.

본 연구결과, 부분점수를 구성에 있어 다음의 내용을 고려해야 할 필요가 있음을 알 수 있다.

첫째, 어떤 오류에 중점을 두고 평가할지에 따라 부분점수를 고려해야 한다. 부분점수를 줄 때, 교사가 평가에서 중요하게 생각하는 오류 종류를 고려하여야 한다.

둘째, 문제 유형에 따른 부분점수 배점 비율을 고려해야 한다. 학생들의 프로그램 오류 분석 결과, 문제 유형에 따라 발생하는 오류 종류에 차이가 있었다. 또한, 설문 결과에서 동일한 종류의 오류라 하더라도 문제 유형에 따라 중요성이 다르게 나타났다. 이를 토대로, 문제 유형에 따라 어떻게 부분점수를 반영할지를 고려하여 부분점수 배점 비율을 고려해야 한다.

5. 결론 및 향후 연구

본 연구에서는 텍스트 기반 프로그램 평가에서 학습을 지원하는 평가를 고려하였고, 정답 여부만을 판단하는 평가보다는 부분점수 구성에 대한 당위성을 확인하기 위해 진행되었다. 목적 달성을 위해 기존의 프로그램 평가 방법들을 분석하고, 수집된 학생의 프로그램의 오류 분석하였다. 분석에 근거하여 교사들의 오류에 대한 의견을 분석한 결과를 논의하면 다음과 같다.

첫째, 학생마다 프로그래밍 개념 및 문법 이해 수준에 따른 개인차로 인해 오류가 발생하는 원인이 다양하게 나타난다. 특히 문제 유형에 따라 발생하는 오류 종류가 다름을 알 수 있다. 따라서 특정 개념에 대한 평가에서는 어떤 오류의 요소들이 자주 발생하는 지를 토대로 향후 수업을 지원하는 데 반영할 필요가 있다.

둘째, 수업 지원의 관점이다. 학생이 자주 실수하는 오류와 교사들이 프로그램 평가에서 중요하게 생각하는 오류가 다르게 나타났다. 학습의 측면에서는 학생이 자주 실수하는 오류를 통해 학생이 이해한 내용과 이해하지 못한 내용을 전반적으로 보다 명확하고 구체적으로 진단하여, 수업에서 지원해야 한다. 그러나 교사들이 중요하게 생각하는 오류에만 집중할 경우, 학생들은 오히려 기본적인 문법이나 들여쓰기와 같은 문법 오류로 인해 프로그램에 대한 흥미를 잃을 가능성이 적지 않을 것으로 판단된다.

셋째, 평가의 관점이다. 교사들은 다양한 오류들 중 중요하게 고려해야 하는 오류에 대해 평가에서도 유사한 생각을 반영할 것으로 응답하였다. 즉, 교사들이 중요하게 생각하는 오류를 고려하여 루브릭을 개발하거나, 부분점수를 주는 방향에 대한 논의가 필요하다.

평가의 관점에서는 동일한 오류 종류에 대해서도 문제 유형에 따라 오류의 중요성이 다르게 나타날 수 있음을 고려해야 한다. 따라서 문제 유형에 따

라 배점 비율을 고려한 부분 점수를 구성할 필요가 있다.

학생이 자신의 생각을 표현하는 도구로 프로그램을 선택했다 해도 프로그래밍 개념과 문법 이해에 수반되는 명령어 암기는 필수적일지 모른다. 그러나 암기에 치중하다 보면, 정보 교과 고유의 가치인 사고력, 표현력 등에 대한 부분이 퇴색할 수 있음을 고려해야 할 것이다. 학생의 사고력을 향상시키는 교육의 가치를 지켜가기 위해서는 프로그램 평가는 명령어 암기와 같은 지식 위주의 평가를 지양하고, 학생이 프로그래밍을 통해 문제를 해결해 가는 과정에 집중할 필요가 있을 것이다.

참 고 문 헌

[1] 우호성 · 김자미 · 이원규 (2017). 해외 고등정보 표준교육과정 기반의 국내 대학 교육과정 비교 분석. **컴퓨터교육학회논문지, 20**(1), 27-38.

[2] 안영희 · 김자미 · 이원규 (2016). 메타학문인 정보학을 모태학문으로 하는 교과. **컴퓨터교육학회논문지. 19**(3), 1-10.

[3] 양혜지 · 김자미 · 이원규 (2018). 해외 정보 교육과정에서 정보문화소양 관련 내용 요소 분석. **컴퓨터교육학회논문지. 21**(3), 1-10.

[4] 윤일규 · 김현철 (2018). 초·중등 학부모의 정보교육에 대한 인식 분석을 통한 정보교과 공교육 정착 방안 탐색. **컴퓨터교육학회논문지. 21**(2), 31-40.

[5] 교육부 (2015). **초·중등학교 교육과정 총론.** 교육부 고시 제 2015-80호 별책 1(교육부 고시 제 2015-74호의 부칙개정).

[6] K-12 Computer Science Framework Steering Committee. (2016). *K-12 computer science framework.*

[7] 김자미 · 이원규 (2014). 브루너의 이론에 근거한 인도의 정보교육과정 고찰. **컴퓨터교육학회논문지, 17**(6), 1-11.

[8] 김자미 · 이원규 (2016). 교육과정 총론의 문서체제에 나타난 고등학교 정보과 교육과정의 변천. **컴퓨터교육학회논문지. 19**(5), 27-40.

[9] 김경규 · 이종연 (2016). 컴퓨팅 사고력 기반 프로그래밍 학습의 효과성 분석. **컴퓨터교육학회논문지, 19**(1), 27-39.

[10] 교육부 (2015). **소프트웨어 교육 운영 지침.** 교육부(교육과정정책과).

[11] 안성훈 (2016). 초·중학교 SW 교육을 위한 프로그램 평가 지표 개발. **컴퓨터교육학회논문지. 19**(4), 11-20.

[12] 김재경 (2017). 컴퓨팅 사고 개념 학습과 프로그래밍 역량 평가를 위한 루브릭 개발. **컴퓨터교육학회논문지. 20**(6), 27-36.

[13] Kurnia, A., Lim, A., & Cheang, B. (2001). Online judge. *Computers & Education. 36*(4), 299-315.

[14] Brito, M. A., & de Sá-Soares, F. (2014). Assessment frequency in introductory computer programming disciplines. *Computers in Human Behavior. 30,* 623-628.

[15] Wang, G. P., Chen, S. Y., Yang, X., & Feng, R. (2016). OJPOT: online judge & practice oriented teaching idea in programming courses. *European Journal of Engineering Education, 41*(3), 304-319.

[16] Jonsson, A., & Svingby, G. (2007). The use of scoring rubrics: Reliability, validity and educational consequences. *Educational research review, 2*(2), 130-144.

[17] 김민자 · 유길상 · 김현철 (2017). 비전공자 프로그래밍 수업 창의적 산출물의 컴퓨팅 사고력 기반 평가 루브릭 개발. **컴퓨터교육학회논문지. 20**(2), 1-11.

[18] 최형신 (2014). Computational Thinking 역량 계발을 위한 수업 설계 및 평가 루브릭 개발. **정보교육학회논문지. 18**(1), 57-64.

[19] Sherman, M., & Martin, F. (2015). The assessment of mobile computational thinking. *Journal of Computing Sciences in Colleges, 30*(6), 53-59.

[20] Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *In Proceedings of the 2012 annual meeting of the American Educational Research Association, 1*, 1-25.

[21] 문외식 (2006). 초등학생들이 프로그래밍 학습

시 발생하는 오류유형 분석. **한국컴퓨터정보학회논문지. 11**(2), 319-327.
[22] 남재원, 유인환 (2011). 오류분석에 기반한 NXC 로봇프로그래밍 지원시스템의 개발. **정보교육학회논문지. 15**(3), 375-385.
[23] 김지선, 김영식 (2014). 온라인 프로그래밍 개념학습 성취수준과 오류유형과의 관계 분석. **컴퓨터교육학회논문지. 17**(5), 43-51.
[24] 장혜선, 최숙경, 전수진, 염용철, 이원규 (2007). 컴퓨터교과교육: 에러 피드백 기반의 초보자를 위한 프로그래밍 학습 지원 시스템. **컴퓨터교육학회논문지. 10**(6), 1-10.
[25] Murray, W. R. (1988). *Automatic program debugging for intelligent tutoring systems.* London: Pitman.
[26] Vipindeep, V., & Jalote, P. (2005). List of common bugs and programming practices to avoid them. *Electronic, March.*

이 재 영

2016 고려대학교 컴퓨터정보학과(공학사)
2019 고려대학교 일반대학원 컴퓨터학과(공학석사)
관심분야: 정보교육, 프로그래밍 교육, 교육평가
E-Mail: jaeyoung.lee@inc.korea.ac.kr

김 자 미

1992 이화여자대학교 교육학과(문학사)
1995 이화여자대학교 교육학과(문학석사)
2011 고려대학교 컴퓨터교육학과(이학박사)
2011 ~ 2015 고려대학교 컴퓨터학과 연구교수
2015 ~ 현재 고려대학교 교육대학원 컴퓨터교육 전공 조교수
관심분야: 정보교육, 교육과정평가, 이러닝
E-Mail: celine@korea.ac.kr

이 원 규

1985 고려대학교 영어영문학과(문학사)
1989 츠쿠바대학 이공학연구과 (공학석사)
1993 츠쿠바대학 공학연구과 전자.정보공학 전공(공학박사)
1993 ~ 1995 한국문화예술진흥원 문화정보본부 책임연구원
1996 ~ 2014 고려대학교 사범대학 컴퓨터교육과 교수
2014 ~ 현재 고려대학교 정보대학 컴퓨터학과 교수
관심분야: 정보교육, 정보표현, 정보관리, 교육정책
E-Mail: lee@inc.korea.ac.kr