엔트리를 이용한 보편적 프로그래밍 언어 개발로 절차적 사고력 향상 방안

김동만† · 이태욱†
† 한국교원대학교 컴퓨터교육과

Enhancement Method of the Procedural Thinking Ability through Universal Programming Language Utilizing Entry

Dong-Man Kim† · Tae-Wuk Lee†
† Dept. of Computer Education, Korea National University of Education

요 약

이 연구에서 엔트리 명령 블록을 이용하여 보편적 프로그래밍 언어를 개발하고 검증하였다. 그래서 이 연구를 통해 블록형 프로그래밍 언어의 접근 수월성을 이용하여 절차적 사고력 향상을 위한 아이디어를 제공하고자 하였다. 새로운 프로그래밍 언어를 만들어 알고리즘을 적용하여 함수화된 사칙연산 프로그램을 만들면서, 다양한 알고리즘을 적용하면 엔트리에서 제시하는 모든 명령 블록을 만들 수 있음을 증명하였다. 이 연구를 통해 1)프로그래밍 언어에 포함된 다양한 기능의 명령어들도 함수화되어 있음을 증명하고 재생산 가능함을 경험할 수 있는 아이디어를 제공하고, 2)초보 프로그래머들이 프로그래밍 언어 개발에 대한 흥미와 관심을 갖게 되는 방안을 제시하며, 3)알고리즘을 경험하면서 절차적 사고력을 향상시킬 수 있는 다른 방향의 SW 교수·학습 방법과, 4)프로그래밍 언어를 미시적으로 탐구하면서 SW 교육 관점을 다양화하는 방법을 제시하였다. 이 연구에서 제시한 방안으로 학생들이 절차적 사고력 향상과 프로그래밍 언어의 다양성 인식, 프로그램을 심층적으로 분석하는 태도 등의 SW 교육에 대한 긍정적 변화를 기대한다.

1. 서 론

오늘날의 여러 고급 언어들에 들어 있는 대부분의 기능은 프로그래밍 언어가 갖는 근본적인 능력에 기여하기보다는 편리한 기능을 강화시키고 있을 뿐이다. 프로그래밍 언어가 간편한 저작도구를 사용하듯 코딩을 배우고 언어 근간의 알고리즘은 깊이 있게 고민하지 않으려 한다. 비의도적이지만 편의성의 이유로 초보 프로그래머들이 프로그래밍 언어 자체에 대한 개념 습득의 기회가 제공되지 않을 수도 있다. 최근 대부분의 고급 프로그래밍 언어가 캡슐화, 모듈화, 함수화 되어 있다. 학생들이 수월하게 배울 수 있도록 한 비주얼 방식의 교육용 프로그래밍 언어(EPL)인 엔트리와 스크래치에서는 이런 경향이 더욱 심화되어 있다. 이런 EPL의 명령 블록을 조립하여 주어진 문제를 해결하면서 고등 사고력을 향상시킬 수 있다고 한다. 절차적 사고로 문제를 최대한 분해하여 절차적으로 코딩하면 쉽게 프로그램을 만들 수 있다. 그런데 초보 프로그래머들이 사고과정 없이 필요한 명령 블록을 찾아 순서대로 끼워 넣는 방법은 단순히 '따라하기'식의 코딩이 될 우려가 있다. 그래서 EPL로 절차적 사고력을 향상시키는 방법에 보충이 필요하다. 절차적 사고력 향상을 위한 방법은 블록 기반 코딩방법보다는 다양한 기능이 빠진 가장 기본적인 텍스트 형식의 프로그래밍 언어를 통해 더 효율적으로 배울 수 있다. 일본에서 개발된 두리틀(Dolittle) 프로그래밍 언어를 한글화 작업을 한 경우도 이 이유이다[1]. 그러나 초보 프로그래머들의 텍스트 방식의 프로그래밍 언어의 거부감 때문에 텍스트 방식의 코딩에 참여하기가 어렵다. 그래서 기존에 익숙한 비주얼 방식의 엔트리 명령 블록을 이용하여 익숙한 코딩 환경을 제공하고 절차지향 언어와 같은 방식의 생각을 하면서 프로그래밍 언어를 분석해 볼 수 있는 경험을 제공하면 절차적 사고력 향상에 크게 기여를 할 수 있을 것이다.

따라서 이 연구로 초보 프로그래머들이 엔트리 명령 블록을 이용한 흥미로운 코딩 도구를 바탕으로 단순하지만 범용적인 프로그래밍 언어의 개발 과정을 경험하면서 절차적 사고력 향상을 위한 SW교육 방법을 제안하고자 한다.

2. 이론적 배경

2.1 튜링 완전한 보편적 프로그래밍 언어

프로그래밍 언어들에서 꼭 필요한 기능이 무엇인지를 확인하기 위해 계산 가능성에 대한 지식을 적용한다. 프로그래밍 언어가 갖는 언어의 근본적인 계산 능력을 확인할 수 있는 것이 중요하다. 보편적 프로그래밍 언어는 계산 가능한 모든 함수를 계산하기 위한 프로그램을 표현할 수 있는 기능을 갖추어야 한다[2].

보편적 프로그래밍 언어는 튜링 완전(Turing Complete)한 프로그래밍 언어라고 할 수 있다. 튜링 완전한 프로그래밍 언어들은 제어구조인 조건 분기문을 가질 수 있고, 메모리의 임의 위치의 값을 변경할 수 있는 기능을 갖고 있다[2]. 이 2가지 기능을 갖고 있으면, 우리가 사용하는 프로그램의 거의 모든 기능들을 2가지 기능의 조합으로 만들어낼 수 있다[2]. 따라서 보편적 프로그래밍 언어를 개발하기 위해서는 제어구조를 사용하면서 메모리를 적절히 사용할 수 있는 기능을 가지고 있으면 된다. 그리고 보편적 프로그래밍 언어인지를 모든 함수를 계산하기 위한 프로그램으로 표현할 수 있는지, 다시 말해 튜링 완전한지 검증해야 한다.

2.2 절차적 사고력

절차적 사고력의 정의는 국내에서 아직 명확하게 정의되지 않고 있다. 그래서 그 의미를 명확히 하고자 비슷한 개념인 절차적 사고와 알고리즘적 사고, 알고리즘적 사고력 등의 개념을 찾아보면, Wing(2006)이 말한 컴퓨팅 사고력의 하위 사고 양식의 하나로 알고리즘적 사고를 제시하였고, 이를 보충하여 이은경(2009)은 인간의 추상화 능력 중 문제 해결 전략에 동시적 사고, 선행적 사고, 전략적 사고, 절차적 사고, 재귀적 사고의 개념을 제시하고 있어서 절차적 사고와 알고리즘적 사고는 컴퓨팅 사고력의 핵심요소임을 알 수 있다[3-4].

Moursund(1997)는 컴퓨팅 사고력의 핵심 사고로 절차적 사고를 강조하면서 컴퓨팅 사고가 문제를 분석하고, 효율적 해결을 위한 순서를 기술하는 절차라고 말하였다[5]. 2015 개정 교육과정 초등 SW영역 성취기준 해설에 따르면, 절차적 사고란 문제를 효율적으로 해결하기 위해 문제를 작은 단위로 나누고, 각각의 문제를 단계별로 처리하는 사고 과정이다[6].

문교식(2013)은 어떠한 문제를 해결하기 위한 논리적인 문제 해결 절차를 알고리즘이라고 할 때, 알고리즘 학습 내용으로 절차적 사고를 제시할 수 있다[7]. 즉 절차적 사고는 알고리즘적 사고와 비슷하거나 이를 포함하는 개념이다.

이태욱 외(2013)는 알고리즘적 사고력은 사고 과정을 안내하는 사고 수단으로 사건 요소 추출과 시간적 질서를 분석하여 알고리즘으로 표현하는 방법으로 향상시킬 수 있다고 하였다[8].

위 선행연구를 통해 절차적 사고력의 의미를 종합적으로 분석하면, 알고리즘을 바탕으로 절차를 중시하면서 일의 상황에 따른 조건을 생각하여 효율적으로 문제를 해결할 수 있는 능력이라 할 수 있다. 문제해결을 위해 알고리즘이 드러나게 문제를 분해하고 각각의 해결방법을 찾아 일련의 절차에 따라 나열할 수 있는 능력을 말하며 컴퓨팅 사고력과 알고리즘적 사고의 핵심 요소라고 할 수 있다.

그래서 절차적 사고력이란, 문제를 해결하는 과정에서 조건을 생각하여 알고리즘이 드러나게 일련의 절차를 나열할 수 있는 사고 능력으로 정의할 수 있다.

3. 연구 설계

3.1 연구 방법

이 연구의 목적이 실제 프로그래밍 언어를 개발하는 과정처럼 최종 소스코드를 생성하는 것은 아니다. 학생 교육 목적으로 기본적인 기능을 포함하여 보편적 프로그래밍 언어가 되기 위한 뼈대를 확인하고 알고리즘으로 프로그램을 제작하여 이를 검증하는 과정을 안내하기 위함이다. 따라서 보편적 프로그래밍 언어의 특징을 분석하고 이를 통해 엔트리를 이용하여 프로그래밍 언어를 개발하고 보편성을 검증하는 과정으로 진행하였다.

이 연구는 엔트리로 보편적 프로그래밍 언어 개발로 절차적 사고력 향상 방안을 모색하는 아이디어 제시가 목적이지만, 후속연구에서 교육 프로그램 개발을 위한 과정을 진행할 것이다. 학생 교육용 프로그램 개발은 ADDIE 모형에 따라 분석, 설계, 개발, 실행을 진행하고 교육의 효과성을 검증 및 평가할 예정이다.

3.2 엔트리를 이용한 보편적 프로그래밍 언어 개발

엔트리 실행 창에 엔트리봇 오브젝트가 보이게 하고 메모리를 적절히 사용하기 위해 변수 4개(A, B, C, D)를 선언한다. 새로 만들 프로그래밍 언어의 절차적 명령 블록들을 살펴보면, 엔트리 프로그램에서 3종류의 배정 명령 블록과 1가지의 제어 구조 명령 블록을 이용한다. 3가지 배정 명령 블록은 변수의 내용을 변경하도록 요청한다. 배정 명령 블록의 종류와 제어 구조를 나타내는 명령 블록은 아래 <표 1>과 같다.

<표 1> 배정과 제어 구조를 위한 엔트리 명령 블록

구분		엔트리 명령 블록
배정 명령 블록	변수 값 초기화 0	변수 값 를 0 로 정하기
	변수 값에 1 더하기	변수 값 에 1 만큼 더하기
	변수 값에 -1 더하기	변수 값 에 -1 만큼 더하기
제어 구조 명령 블록	변수 값이 0보다 큰 동안 반복	변수 값 > 0 인 동안 반복하기

첫 번째 형식의 배정 명령 블록을 사용하여 변수의 값을 초기화, '0'으로 만들 수 있다. 두 번째와 세 번째의 배정 명령 블록은 각각 변수 값에 1과 -1을 더하는 명령 블록이다.

엔트리에서 이용 가능한 제어 구조 중, 단 한 가지 while문의 명령 블록만 이용한다. 프로그램 실행 중에 제어 구조를 만나면 0보다 큰지를 비교하여 조건에 맞는 만큼 반복을 실행한다.

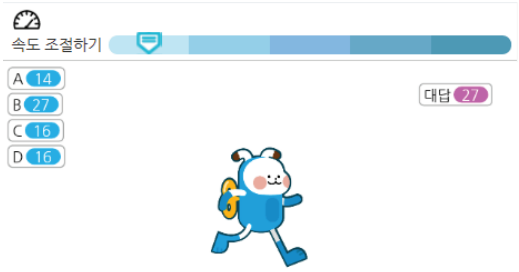
엔트리 프로그램에서 3개의 배정 명령 블록과 제어 구조를 위한 1개의 명령 블록, 총 4개의 명령 블록으로 튜링 완전한 조건을 갖추게 된다. 학생 교육 프로그램을 개발할 때 다양한 배정 명령 블록과 if문과 같이 제어를 위한 다른 명령 블록을 이용하여 튜링 완전한 조건을 탐색해볼 수 있도록 할 수 있다.

이렇게 보편적 프로그래밍 언어의 개발과정에서 단 4개의 명령 블록으로 강력한 프로그램을 만드는 힘은 바로 절차적 사고력을 통해 가능함을 배운다. 따라서 언어의 개발 과정을 경험하면서 유명 프로그램도 특정 프로그래밍 언어의 기능이 대단해서라기보다는 인간의 사고 과정인, 절차적 사고력이 더 중요함을 인식하게 된다.

3.3 보편적 프로그래밍 언어의 검증

엔트리로 개발한 언어가 보편적 프로그래밍 언어임을 검증하기 위해서는 튜링 논제를 적용한다. 보편적 프로그래밍 언어가 계산 가능한 모든 함수를 계산하기 위한 프로그램을 표현할 수 있는 기능을 갖추고 있는지를 검증하면 된다[2]. 그리고 컴퓨터 프로그래밍은 효율성을 바탕으로 최적화 과정을 거치는데 필요한 알고리즘을 경험해야한다.

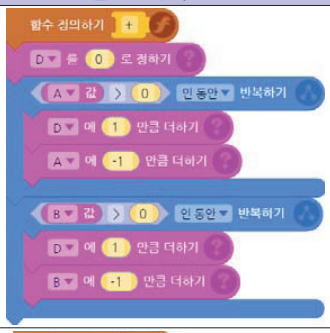
각 함수를 계산하기 위해 변수 값을 적절히 지정하기 위한 배정 명령 블록으로 시작되는, 함수로 제작된 프로그램을 실행시킨다. 그 다음 프로그램이 종료될 때 변수들의 값을 관찰하면 된다. 엔트리 실행장면은 [그림 1]과 같다.



[그림 1] 엔트리 실행 장면

함수로 만들어진 프로그램 코드는 <표 2>와 같다.

<표 2> 엔트리 함수 사칙연산 프로그램

구분	엔트리 함수 코드
덧셈	
뺄셈	
곱셈	
나눗셈	

이와 같이 함수로 만들어진 프로그램의 입력은 프로그램의 실행 전에 입력받아 변수 A와 B에 배정된 값으로 이루어지며, 함수의 출력은 프로그램이 종료되는 시점에 변수 D에 배정되어 있는 값으로 이루어진다. 이러한 조건 하에서 함수로 제작된 프로그램을 각각 실행해 보면 A와 B의 값을 입력받아 원하는 사칙연산이 가능한 알고리즘을 통해 처리되어 출력되는 D값을 확인하면 된다. 위의 코드들을 확인했을 때 계산 가능한 임의의 함수는 새로 개발된 언어로 작성된 프로그램에 의해 계산됨을 확인할 수 있었다.

따라서 이렇게 만들어진 프로그래밍 언어는 보편적 프로그래밍 언어이기 때문에 위와 같은 사칙연산이 가능한 프로그램 외에도 이론적으로는 엔트리에서 제시하는 모든 명령 블록을 함수로 만들 수 있다. 이론적이라는 의미는 알고리즘이 존재한다고 가정한다면의 의미로 C++이나 Java와 같은 고급 언어처럼 범용 프로그래밍 언어로 사용가능하다는 것이다.

4. 결론 및 제언

이 연구는 엔트리 명령 블록을 이용하여 새로운 프로그래밍 언어를 개발하고 검증하였다. 그리고 새로운 프로그래밍 언어로 알고리즘을 적용하면 엔트리에서 제시하는 사칙연산 명령 블록을 본 연구에서 제시한 4개의 명령 블럭의 조합으로 만들어낼 수 있었다. 그래서 이 연구의 결론은 다음과 같다.

첫째, 학생들이나 초보 프로그래머들에게 프로그래밍 언어에 포함된 다양한 기능의 명령어들도 함수화되어 있음을 증명하고 재생산 가능함을 경험할 수 있는 아이디어를 제공한다.

둘째, 엔트리 기반의 명령 블록을 이용하여 보편적 프로그래밍 언어의 제작 가능한 방법을 경험하면서 초보 프로그래머들이 프로그래밍 언어 개발에 대한 흥미와 관심을 갖게 되는 기회가 된다.

셋째, 프로그램 언어 개발과정에서 알고리즘을 경험하면서 절차적 사고력을 향상시킬 수 있는 다른 방향의 SW 교수·학습 방법을 제공한다. 엔트리에서 제공하는 다양한 명령 블록을 최소한의 명령 블록으로 만들어보면서 비주얼 방식 프로그래밍 언어의 명령 블록 속에 숨겨진 알고리즘을 개발하면서 절차적 사고력을 향상시킬 수 있다.

넷째, 프로그래밍 언어를 미시적으로 탐구하면서 SW 교육 관점을 다양화하는 방법을 제시한다. 명령 블록이나 명령문의 생성과정을 구체적으로 알아보면서 다양한 프로그래밍 언어에 대한 심도있는 관찰로 이어질 것이다.

이 연구에서 제시한 방안으로 학생들이 프로그래밍 언어가 갖는 기본적인 뼈대를 이해하고, 알고리즘의 중요성을 인식하며 절차적 사고력 향상과 프로그래밍 언어의 다양성 인식, 프로그램을 심층적으로 분석하는 태도 등의 SW 교육에 대한 긍정적 변화가 생기길 기대한다.

참 고 문 헌

[1] 천준석, 강도훈, 김건우, 우균(2015). 간결한 한글 프로그래밍 언어 “새싹”. **정보과학회 논문지, 42**(4), 496-503.

[2] J. Glenn Brookshear & Dennis Brylow(2015). ***Computer Science An Overview***. Pearson Education.

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

[4] 이은경(2009). **Computational Thinking 능력 향상을 위한 로봇 프로그래밍 교수 학습 모형**. 한국교원대학교 박사학위논문.

[5] Moursund, D.(1977). ***Computational Thinking and Math Maturity: Inproving Math Education in K-8 Schools***. Eugene, Oregon: University of Oregon Press.

[6] 교육부(2015). **실과(기술・가정)/정보과 교육과정**. 교육부 고시 제2015-74호 [별책10].

[7] 문교식(2013). 계산사고의 적용을 위한 알고리즘 학습. **한국정보교육학회 논문지. 제4권,** 295-300.

[8] 전수련, 남동수, 이태욱(2012). 초등 정보영재의 알고리즘적 사고력 향상을 위한 실생활 주제의 이야기 쓰기 교수.학습 프로그램. **한국컴퓨터정보학회 학술대회 논문집, 20**(1), 119-122.