프로그래밍 학습에서 학습자의 어려움 분석

최정원† · 이영준††

요 약

프로그래밍은 아이디어를 실현시키는 데 매우 훌륭한 도구이다. 그러나 학습자들은 프로그래밍이 요구하는 엄격한 문법과 고도의 추상적인 사고로 인하여 학습의 어려움을 호소하는 경우가 많다. 학습자들이 쉽게 프로그래밍을 학습할 수 있도록 다양한 연구가 진행되고 있지만 그 이전에 학습자들이 무엇을 어려워하는지에 대한 분석이 선행되어야 학습 효과를 보다 높일 수 있다. 따라서 본 연구에서는 프로그래밍 기초 교육에서 초보 학습자들이 어려워하는 부분은 무엇인지, 그 이유는 무엇인지에 대하여 프로그래밍 개념을 중심으로 분석하였다. 분석 결과를 토대로 프로그래밍 교육 시 프로그래밍 개념에 대한 정확한 이해와 내면화, 정교한 알고리즘 작성을 통해 사고와 실행 결과 간의 오차 축소, 다양한 문제 해결 경험 제공을 바탕으로 문제 해결 능력을 향상시키고, 문제 해결 과정에 대해 자유롭게 사고 할 수 있는 전략을 수립하며, 학습 순서 설계 등을 통한 학습의 효율성을 촉진시킬 필요가 있음을 제시하였다.

주제어 : 프로그래밍 교육, 프로그래밍 학습의 어려움, 컴퓨팅 사고력

The analysis of learners' difficulties in programming learning

JeongWon Choi† · YoungJun Lee††

ABSTRACT

Programming is excellent tool on realizing ideas. However students often complain of difficulties due to requiring the strict programming grammar and the highly abstract thinking. Although various researches have been conducted to teach the programming easily for students, it should precede the analysis of what and why programming concept is difficult for learners. In this study, we analyzed what and why the programming concept is difficult for novice learners in basic programming education. Based on the results, we suggested: improving problem-solving skills based on accurate understanding and internalization on the programming concept, on reducing error between thought and execution results through the creation of sophisticated algorithms and on offering a variety of troubleshooting experience, establishing strategies to think freely for problem-solving process, and promoting the effectiveness of the learning through the learning procedure design.

Keywords : Programming Education, Difficulties of Programming Learning, Computational Thinking

† 종신회원: 한국교원대학교 컴퓨터교육과 박사과정
†† 종신회원: 한국교원대학교 컴퓨터교육과 교수(교신저자)
논문접수: 2014년 8월 29일, 심사완료: 2014년 9월 18일, 게재확정: 2014년 9월 26일
* 본 논문은 2012년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2012R1A1A4A01019396)
* 본 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2013R1A2A2A03068459)

1. 서론

21세기 사회는 실생활 뿐 아니라 다양한 학문 분야의 문제를 효율적으로 해결하기 위하여 컴퓨팅 시스템을 다루는 것이 필수인 시대가 되었다. 프로그래밍은 사람의 아이디어를 컴퓨팅 시스템이 실행하도록 함으로써 실천 가능한 형태로 만들어 주기 때문이다. 주요 선진국이나 IT 강국들은 프로그래밍의 중요성을 일찍이 간파하고 국가 차원의 교육과정을 ICT 활용 중심에서 프로그래밍 교육 중심으로 개편・실천하고 있다[1][2][3].

그러나 프로그래밍은 프로그래밍 언어의 엄격한 문법 학습 혹은 고도의 추상화 능력을 필요로 하기 때문에 인지적 부담을 증가시켜 학습자가 중도에 포기하도록 하는 결과를 낳는 경우가 많다[4][5].

물론 프로그래밍 교육의 효과를 높이기 위하여 교육용 프로그래밍 언어를 활용하고 학습자들에게 흥미를 부여함으로써 학습 참여를 촉진시키는 사례들도 있다[6][7]. 또한 학습자의 학습 양식을 분석하여 개별화된 교육을 실시하거나 프로그래밍 교육의 효과를 높이기 위한 새로운 교육 방법을 탐색하는 등 보다 쉽게 가르치고자 하는 경우도 있다[5][8][9]. 그러나 프로그래밍의 어려움을 호소하는 학습자들은 여전히 존재하며, 특히 초보자들의 경우가 그러하다[10][11].

초보 학습자들의 프로그래밍 교육을 위해서는 우선 적절한 프로그래밍 도구가 선정이 되어야 하며, 프로그래밍 언어의 문법 학습 보다는 문제해결에 필요한 개념을 학습하고 개념을 적용하는 원리를 익히는 데에 중점을 두어야 한다. 이를 바탕으로 학습자들이 어려워하는 점은 무엇인지, 그 이유는 무엇인지에 대해 파악함으로써 학습의 효과를 높일 수 있는 전략을 수립할 수 있다. 이러한 분석은 학습 과정에서 오개념이 형성되는 것을 예방할 수 있으며, 학습 동기와 흥미를 유지시키고 프로그래밍 학습이 지속되기 위한 적절한 교육적 처방이 가능하도록 한다.

따라서 본 연구에서는 초보 학습자의 프로그래밍 교육에 적절한 '코드닷오알지'의 학습 단계를 수업에 도입하고, 학습자들이 어려워하는 프로그래밍 개념이나 원리, 어려운 이유에 대해 분석한 후 이를 바탕으로 적절한 교육 방안을 제시하였다.

2. 이론적 배경

2.1 프로그래밍 교육

프로그래밍은 상상했던 추상 모델을 현실로 이어주는 다리 역할을 하며, 현실화된 프로그램을 많은 사람들이 공유함으로써 다양한 부가 가치를 창출하고 삶의 편리를 가져다주고 있다.

그러나 프로그래밍 학습은 프로그래밍 언어의 활용을 위한 문법 학습이 선행되어야 하며, 문법 학습과 더불어 추상적인 사고력의 요구로 인해 많은 학습자들이 어려워하고 중도에 포기하기도 한다[4][5].

이러한 어려움을 극복하기 위해 다양한 연구들이 진행되어 왔다. 학습자의 초기 학습 동기를 지속적으로 유지시키기 위해 프로그래밍 교수 학습 환경과 학습자들의 특성을 고려하여 적절한 프로그래밍 교수 학습 모형과 전략을 개발하거나, 학습자의 대부분이 흥미를 갖는 게임을 프로그래밍 교육에 접목하여 게임 프로그램을 개발하도록 함으로써 자기효능감을 향상시켜 프로그래밍 학습을 지속시키고자 하는 연구도 진행되었다[5][8][9].

다양한 교육 방법의 도입이 프로그래밍 학습의 향상에 긍정적인 영향을 미치고 있지만, 실질적으로 학습자들이 어떠한 어려움을 겪고 있는가에 대한 분석 없이는 제시된 방법 또한 프로그래밍 학습의 효과를 최대화시키기는 어렵다.

Milne과 Rowe(2002)은 학습자들이 프로그래밍 학습에서 어떠한 어려움을 겪는지 확인하기 위하여 대학생 학습자들을 대상으로 C와 C++ 프로그래밍을 교육하고 설문을 실시하였으며 포인터와 메모리 할당 부분을 학습자들이 가장 어려워한다는 것을 확인할 수 있었다. 그리고 그 원인이 학습자들이 프로그램 실행 동안 메모리에서 어떤 작업이 수행되고 있는지에 대해 충분히 이해하지 못했기 때문이라고 하였다[12].

Piteira와 Costa(2013)는 프로그래밍이 갖는 교육적 가치에 비해 많은 초보 학습자들이 어려움을 호소하고 있기 때문에 그 어려움이 무엇인지

를 파악해야 한다고 주장하였다[10]. 이들은 어려움을 겪을 수 있는 범주를 학습 개념, 학습 환경, 교육 방법, 교육 주제, 교육 자료로 구분하고 교사와 학습자의 의견을 수렴하였는데, 분석 결과 학습자들은 프로그래밍 문법 학습에 있어서는 포인터와 참조, 매개변수, 구조화된 데이터 타입, 에러 핸들링, 라이브러리 활용에, 교육 방법으로는 혼자 학습하는 부분을 더욱 어려워 한 것을 확인할 수 있었다. 학습 자료의 경우는 코드 예제와 인터넷 튜토리얼, 교육용 동영상, 무들 플랫폼에서 활용 가능한 콘텐츠 순으로 학습하기 용이한 자료로 나타났다.

그러나 선행 연구에서 사용된 학습 도구들은 텍스트 기반의 언어로 문법 구조와 컴퓨팅 개념 및 원리 활용 등에 있어 엄격하기 때문에 초보 학습자들에게는 인지적인 부담을 증대시킬 수 있다는 점에서 기초 단계의 학습에 도입하기에 적절하지 않다. 또한 학습자가 어려워하고 있는 부분이 무엇인지만 제시할 뿐 왜 어려워하는지에 대한 원인 파악이 정확하게 이루어져 있지 않다. 따라서 초보 학습자들에게 문법 학습에 대한 인지적 부담을 주지 않으면서 개념과 원리를 학습시기에 적절한 프로그래밍 언어를 도입하고, 학습 과정에서 겪는 어려움은 무엇인지, 그 원인은 무엇인지를 진단하여 프로그래밍 학습을 향상시키기 위한 적절한 처방을 내릴 필요가 있다.

2.2 코드닷오알지

코드닷오알지(http://code.org)는 미국의 코드닷오알지라는 비영리단체가 프로그래밍 교육의 확산과 의무화를 위하여 시작한 범국가적인 캠페인이다. 모든 사람에게 프로그래밍 학습의 기회를 제공하는 것을 목적으로 하기 때문에 재미있는 동영상과 캐릭터로 컴퓨터 과학 분야를 이해하고 프로그래밍을 학습하도록 한다.

코드닷오알지 교육 프로그램의 특징을 살펴보면 다음과 같다.

첫째, 코드닷오알지의 학습 과정은 총 20 단계로 구성되어 있다. 오프라인 활동을 할 수 있는 워크북과 온라인에서 프로그래밍을 경험할 수 있는 퍼즐을 포함한다.

워크북을 활용한 오프라인 활동에서는 컴퓨터 과학에 대해 이해하고 학습의 필요성을 인식하도록 한다. 반면에 온라인 활동에서는 오프라인으로 학습한 프로그래밍의 개념을 학습하고 개념을 적용하여 캐릭터가 주어진 미션을 수행하도록 코드 블록을 조합함으로써 프로그래밍의 실제를 경험하도록 하고 있다.

둘째, 학습의 흥미를 향상시키고 적극적인 참여 유도를 위한 다양한 전략이 활용된다. 학습 초기 프로그래밍 개념을 설명하는 단계는 대통령, 컴퓨팅 분야의 유명 인사, 연예인, 잘 알려진 운동선수 등이 동영상에 등장하여 실생활의 사례와 연결시켜 설명함으로써 쉽게 이해할 수 있도록 한다. 이들은 개념이 포함된 코드 블록을 드래그앤드롭 방식으로 연결하여 프로그램을 구성할 수 있다는 프로그래밍 방법을 알려주며 이를 바탕으로 프로그래밍을 경험하도록 한다.

뿐만 아니라 학습 과정에서 학습해야 할 개념에 대한 재설명이 필요하거나 문제 해결에 필요한 힌트를 제공 받고 해법을 확인할 수 있으며 성공에 대한 보상으로 트로피를 받게 된다.

각 학습 단계별로 학습자들의 사고가 반영된 코딩 결과를 캐릭터의 움직임을 통해 확인할 수 있으며, 캐릭터의 움직임에 따라 현재 실행되는 코드 블록에 노란색 띠를 둘러 표시해주기 때문에 코드 블록 중 오류가 있는 부분을 파악하기 쉽도록 하고 있다.

셋째, 체계적인 학습 단계를 바탕으로 다양한 학습 방법을 활용한다. 코드닷오알지는 프로그래밍 개념 학습 - 개념의 활용 학습 - 다양한 개념을 응용하는 심화 학습의 3단계 구조를 바탕으로 프로그래밍 개념, 프로그램 작성법, 프로그램 해석, 디버깅을 학습하도록 한다.

미션을 수행하는 과정에서 주어진 코드 블록을 해석하고 필요한 블록을 삽입하기, 주어진 코드 블록 해석 후 오류 수정하기, 모든 미션 수행 과정 스스로 작성하기, 코드 블록 해석하고 결과 예측하기 등의 방법을 다양하게 도입하여 학습 단계별로 학습자가 인지적인 부담을 느끼지 않으면서 지루하지 않도록 하고 있다.

3. 연구 방법

3.1 연구 대상

K 대학에서 컴퓨팅을 이해하고 활용하는 능력을 길러주기 위한 목적으로 개설된 교양 과목을 수강하는 학생들을 대상으로 12차시 진행하였다. 수강한 학생들은 프로그래밍 경험이 전혀 없는 1, 2학년 학생 20명이며, 이 중 코드닷오알지 퍼즐을 모두 해결하지 못하고 중도에 포기한 학습자 3명을 제외한 17명(남학생 10명, 여학생 7명)의 문제 해결 과정을 분석하였다.

3.2 연구 방법

학습자들이 코드닷오알지에 포함된 20개의 학습 단계 중 실제로 프로그래밍을 하는 9개의 단계에서 제시된 퍼즐을 해결하도록 하였다. 9개의 단계에는 총 98개의 퍼즐이 있으며 그 중 3문항은 원하는 대로 자유롭게 코딩해보는 문항으로, 학습자들이 계획 없이 코딩했다고 응답한 사례가 많아 이를 제외한 95문항에 대해 분석하였다.

학습자들에게는 원하는 결과를 얻지 못했거나 혹은 자신의 코딩에 확신이 없었던 경우, 이해가 잘 되지 않았던 부분에 대해서 그 이유와 함께 구체적으로 작성하도록 안내하였다.

분석은 코드닷오알지에 포함된 프로그래밍 개념을 순차, 반복, 조건, 이벤트로 구분하고 학습자별로 어떤 개념과 원리를 어려워하였는지, 왜 그러했는지를 분석하였다.

4. 연구 결과

4.1 순차

순차는 문제 해결에 필요한 코딩 블록을 해법에 접근 가능한 구체적인 순서에 따라 배치하는 개념이다. 앞서 작성된 코드 블록들의 의미를 바탕으로 현재 삽입되어야 하는 코드 블록이 무엇인지를 정확하게 찾아 절차적으로 나열해야 하며, 순서가 올바르지 않으면 잘못된 결과를 산출하게 된다.

4.1.1 문제 해결 과정 분석

학습자들은 문제 해결에 필요한 블록을 해결 절차대로 배치하지 못하고 뒤바꾸어 배치하거나, 필요한 블록 생략 혹은 불필요한 블록의 추가 삽입, 잘못된 블록의 배치 등의 어려움을 겪은 것으로 나타났다.

이러한 경우는 앞으로 이동하기 블록과 좌우로 회전하기 블록의 배치 순서를 뒤바꾸어 놓거나, 삼각형 두 개를 그리는 경우 하나의 삼각형을 완성한 후 나머지 삼각형을 그릴 때 필요한 각도 회전 블록을 생략하거나 부적절한 각도를 입력한 사례 등이 이에 속한다.

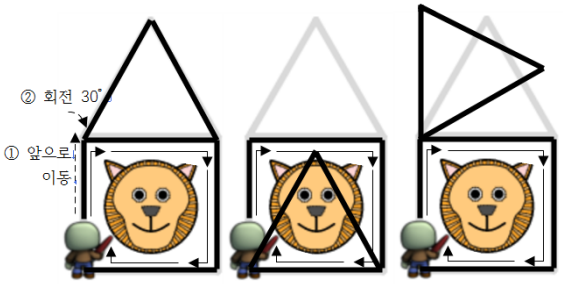
4.1.2 원인 분석

첫째, 학습자들은 체계적으로 작성한 알고리즘을 바탕으로 퍼즐을 해결하는 대신 즉흥적으로 사고한 방법을 바탕으로 프로그래밍 개념을 적용하고 있었다. 순차는 설계한 알고리즘을 바탕으로 필요한 블록을 순서대로 나열할 때 문제 해결에서의 어려움을 최소화시킬 수 있다. 그러나 알고리즘을 구체적으로 설계하지 않음으로 인하여 필요한 블록을 생략하거나 잘못된 위치에 삽입하거나 불필요한 블록을 삽입함으로써 문제 해결의 어려움을 경험하게 되었다.

둘째, 문제 분해 후 통합하는 과정에서 필요한 블록을 누락시키고 있었다. 문제를 작은 문제로 분해하여 해결하고 다시 합치는 방법은 문제를 쉽게 해결하도록 하는 방법이지만, 분해했던 작은 부분을 하나로 합치는 과정에서 작은 부분을 생략해서는 안되며, 작은 부분들을 연결하기 위해 필요에 따라 추가 블록을 삽입할 수 있어야 한다.

한 예로 삼각형과 사각형을 합쳐 집 모양을 그리는 경우를 생각해보자. 삼각형을 그리는 패턴과 사각형을 그리는 패턴을 분해하여 각각에 대한 해법을 생각한 후 합치는 과정에서, 사각형을 그린 다음 삼각형을 그리려면 사각형 그리기가 종료된 지점에서 적절한 이동과 각도의 회전이 추가되어야 한다. 그러나 학습자들은 이동 거리와 회전각을 잊은 채로 삼각형의 한 변을 그리는 작업을 바로 나열함으로써 원하는 결과를 얻지 못

하게 된다. 아래의 <그림 1>에서와 같이 a는 문제 분해 후 필요한 블록(앞으로 이동, 회전)을 추가함으로써 통합이 잘 된 사례이며, b는 추가적인 이동이 생략되어 오류가 난 사례, c는 필요한 회전각도 입력이 생략된 사례를 나타낸다.



a b c

<그림 1> 도형 그리기 수행 예시

셋째, 컴퓨터가 문제를 해결하는 관점에서 사고하지 못했다. 코드닷오알지에서는 컴퓨터가 문제를 해결하는 관점을 가시화시키기 위하여 캐릭터를 도입하고 있다. 문제를 해결하기 위해서는 이 캐릭터의 현재 방향과 위치를 기준으로 어느 방향으로 회전해야 할지, 몇 도의 각도로 회전해야 할지, 앞으로 이동해야 하는 거리는 얼마나 되는지를 분석하고 적합한 블록을 제시할 수 있어야 한다. 그러나 학습자들은 캐릭터의 입장에서 수행해야 할 동작을 분석하지 않음으로써 다음에 배치되어야 할 코드 블록을 잘못 선택하거나 생략하는 어려움을 겪고 있었다.

사람은 사고결과를 행동으로 옮기는 과정에서 신체를 어떻게 움직여야 할지에 대한 구체적인 계획을 세우지 않는다. 신체의 동작은 주로 체화되어 나타나는 경우가 많기 때문이다. 예를 들어, 사람이 길을 찾아갈 때 갈림길에서 목적지 도달을 위한 방향 선택은 심사숙고의 과정을 통해 결정하지만, 선택한 방향을 향해 몸을 회전시키고 걸어가는 행동은 대부분 무의식적으로 이루어진다. 그러나 컴퓨터가 수행하는 방법은 사람과는 다르다. 컴퓨터는 이동해야 할 방향이 결정되고 나면 그 방향을 가기 위해 회전해야 할 각도, 이동 거리에 대한 정보를 습득하고 이를 제시해야 정확히 수행할 수 있다.

넷째, 문제 해결에 필요한 자료를 정확히 수집하지 못한 데 있다. 사람은 해결해야 할 문제에 직면했을 때 문제와 관련된 정보들을 수집함으로써 해결의 실마리를 찾는다. 코드닷오알지는 각 단계 퍼즐별로 제시되는 문제나 캐릭터의 활동 무대가 되는 이미지에서 문제 해결의 단서를 제공하고 있다. 예를 들어 여러 개념들을 필요로 하는 복잡한 문제의 경우, 문제 해결에 반드시 포함되어야 하는 개념의 정의를 다시 제시한다거나 문제 해결에 특정 블록을 사용할 것을 권장하는 등의 힌트가 제공된다. 또한 캐릭터가 움직이는 무대 이미지에서는 캐릭터가 현재 어느 방향을 보고 있는지, 캐릭터가 움직여야 하는 이동 거리는 얼마나 되는지, 어떤 작업을 수행해야 하는지에 대한 단서를 제공한다. 그러나 학습자들은 이러한 단서들을 놓침으로써 순차적으로 실행되어야 하는 블록을 생략하는 경우가 많았다.

다섯째, 과거 학습 경험이 현재의 학습에서 인지적 조작을 방해하기도 한다는 점이다. 과거 학습 경험에 대한 고착 현상은 학습자의 사고가 유연하게 이루어지지 못하도록 한다[13]. 예를 들어, 이동과 각도 블록을 활용하여 사다리를 그리는 경우, 캐릭터가 한 번 지나갔던 길은 다시 가면 안 된다는 과거 한붓그리기의 조건을 떠올려 적절한 블록을 삽입하지 못한 사례들이 이에 해당한다. 또한 삼각형 한 내각은 60°라는 학습 경험이 삼각형을 그리기 위해 캐릭터가 회전해야 하는 각도마저도 60°라고 인식하도록 했다. 이러한 부분은 학습자들이 절차적으로 나열해야 하는 정확한 블록 대신 잘못된 블록을 제시하도록 하였다.

4.2 반복

반복은 문제 해결 과정에서 연속적으로 나타나는 패턴을 인식하고 이 패턴이 반복 실행되도록 구성하는 개념으로, '횟수', '할 때까지', '인 동안', '카운트', '반복의 중첩'이라는 5가지 종류로 나뉜다. '횟수'는 입력한 횟수만큼 주어진 블록 반복하는 개념이며, '할 때까지'는 횟수에 관련 없이 제시한 목적에 달성할 때까지, '인 동안'은 조건이 참인 동안 주어진 블록을 반복하도록 한다. '카운

트'는 시작 수치부터 종료 수치에 도달할 때까지 일정 수만큼 반복적으로 증가 혹은 감소시키고 그 증감되는 수를 변수로 활용하는 개념이다. '반복의 중첩'은 반복 블록 내에 또 다시 반복 블록이 삽입되는 개념이다.

4.2.1 문제 해결 과정 분석

학습자들은 전반적으로 반복 블록 내에서 패턴으로 구성되는 블록들을 순차적인 형태로 중복하여 나열하거나 불필요한 블록을 삽입하였으며, 심지어는 일부 반복 블록이 반복 기능을 갖고 있다는 사실을 고려하지 못했던 것으로 나타났다. 뿐만 아니라 반복되는 횟수를 잘못 입력하거나 문제 해결에 적합한 반복 개념을 선택하지 못하는 경우도 있었다. 또한 반복 중첩에 있어서 두 반복 개념을 병렬적으로 나열하거나 중첩 자체를 떠올리지 못하였으며, 블록의 내포 관계를 정확하게 파악하지 못하기도 하였다.

4.2.2 원인 분석

첫째, 학습자들이 반복 블록의 각 개념과 원리를 정확하게 파악하지 못했다는 점이다. 블록이 반복 기능을 가지고 있다는 것을 인식하지 못하거나 4가지의 반복 개념이 어떤 차이를 갖고 있는지를 파악하지 못한 채 동일한 반복을 수행한다는 수준의 추상적 이해는 반복 블록을 필요한 위치에 정확히 배치하지 못하는 어려움을 겪도록 한다.

또한 개념을 이해하고 있어도 반복 블록을 어디에 어떻게 적용해야 하는지가 익숙하지 않으면 프로그램을 완성하기 어렵다. 패턴은 반복 블록 내부에 삽입되어야 한다는 것을 알지 못하거나 반복 횟수를 잘못 지정하는 등의 학습자가 경험한 어려움이 이에 속한다.

둘째, 반복문 내에 삽입되어야 할 반복되는 패턴을 정확하게 인식하지 못했다. 패턴을 인식하지 못하면 효율적인 프로그램을 완성할 수 없으며, 정확하게 파악되지 못한 패턴은 반복문 내에 포함되어야 하는 블록이 생략되거나 불필요한 블록을 포함하게 됨으로써 문제 해결에 오류를 만들어 낸다.

셋째, 중첩의 구조화에 대한 충분한 파악이 부족하다. 이는 사람의 사고 과정이 순차적이기 때문에 초보 학습자들은 구조화에 집중하지 않으면 순차 구조로 표현하기 쉽다. 이러한 부분은 문제 분해 전략을 사용한 문제 해결 과정에서도 나타난다. 문제 분해 후 다시 합칠 때 의도적으로 중첩을 구조화가 필요한지 확인할 필요가 있으며, 알고리즘 설계 시 포함 관계를 명확히 하도록 하고 정교하게 설계함으로써 어려움을 겪지 않도록 예방할 필요가 있다.

넷째, 문제 해결 과정이 복잡해질수록 학습자들의 문제 해결 패턴이 주로 무조건적인 시행착오 접근 방법에 집중되어 있다. 중첩의 등장으로 복잡도가 높아지면서 학습자들은 프로그램을 계획하려는 노력 대신 일단 실행시켜보고 수정하는 시행착오 방법을 통해 문제를 해결하고 있었다. 이러한 점은 학습자들이 중첩을 사용하면 문제를 효과적으로 해결할 수 있다는 인식 뿐 아니라 어떤 블록이 내포되는 블록이고 어떤 블록이 다른 블록을 내포하는지 구조화 시키는 방법을 깨닫기 어렵도록 한다.

4.3 조건

'조건'은 조건의 가부를 바탕으로 특정 동작이 발생하도록 분기가 이루어져야 할 때 사용하는 개념이며 '만약'과 '만약, 그렇지 않다면'의 두 가지 블록을 포함한다.

4.3.1 문제 해결 과정 분석

학습자들은 '만약' 블록이 반복 개념이나 회전의 기능을 포함하고 있다고 인식하거나 조건 블록 내에 어떤 블록이 포함되어야 하는지를 파악하지 못하는 어려움을 경험하였다.

4.3.2 원인 분석

첫째, 학습자들이 조건에 대한 구조와 적용 원리를 정확하게 파악하지 못했다는 점이다. 조건 블록은 문제를 해결하는 과정에서 선택이 필요한 경우 현재 조건 상태에 따라서 다음에 발생할 선

택 동작을 구성하도록 되어 있다. 이 개념을 적용하기 위해서는 선택을 위한 조건을 선언하고 조건에 따라 발생할 동작을 블록 내부에 작성하게 하는데, 학습자들은 조건 선언 부분에 이미 특정 동작 블록 기능(예: 왼쪽으로 회전하거나 오른쪽으로 회전하는 등)이 함께 포함되어 있다고 인식하고 있었기 때문에 조건에 따른 수행 블록을 어떻게 구성해야 할지 판단하지 못했다.

둘째, 코드닷오알지의 학습 순서에서 그 원인을 찾을 수 있다. 코드닷오알지의 학습 과정은 반복 개념 학습 후 조건 개념을 학습하도록 한다. 이로 인해 학습 순서는 학습자들이 학습할 개념에 앞서 이전에 학습한 내용의 영향을 받는 경우가 있다. 학습자들이 반복 개념의 학습 경험이 조건 블록에도 반복의 기능을 포함되어 있다고 인식하도록 하는 경우가 많았다.

4.4 이벤트

이벤트는 한 동작이 다른 동작의 발생을 위해 프로그램의 실행 제어권을 넘기는 개념으로, 코드닷오알지에서 제시하는 이벤트는 함수이다. 함수는 문제 해결 과정 전반에 걸쳐 반복 사용될 수 있는 모듈로 주프로그램과 별도로 작성된다. 순차적으로 실행되는 주프로그램 내에서 함수 모듈이 필요한 상황(이벤트)이 발생하면 호출을 통해 사용한다.

함수 호출은 주로 주프로그램에서 전달하는 매개 변수와 함께 사용되는 경우가 많은데, 문제 해결 패턴은 동일하지만 처리해야 할 자료가 다른 경우 그 자료를 매개 변수를 통해 함수의 입력으로 받아 처리할 수 있다.

4.4.1 문제 해결 과정 분석

학습자들은 주로 함수를 사용하지 않고 필요한 블록을 반복해서 절차적으로 나열하거나 반복문을 사용하여 해결하려고 하는 경향을 보였으며, 함수를 호출해야 하는 위치나 함수에 포함되어야 하는 블록을 정확하게 판별하지 못한 데에서 어려움을 겪었다. 또한 매개 변수가 삽입되어야 할 위치를 정확하게 파악하지 못하는 어려움을 경험한 것으로 나타났다.

4.4.2 원인 분석

첫째, 함수 개념에 대한 정확한 이해가 부족했다. 개념에 대한 이해는 그 개념이 왜 필요한지, 어떤 역할을 하는지를 깨닫게 한다. 그러나 학습자의 이해가 반복문을 하나의 모듈로 분리시켜놓았다는 수준에 그침으로써 그 필요성 인식이나 정확한 개념 파악이 되지 않은 경우가 많았다. 학습자들의 반응 중 함수 대신 문제 해결에 필요한 블록을 절차적으로 나열하였거나 함수와 반복문의 차이를 인식하지 못하고 반복문으로 문제를 해결하는 사례들이 이에 해당된다.

함수 개념에 대한 이해 부족은 매개 변수의 사용 방법 또한 익히지 못하도록 한다. 매개변수의 경우는 독립적으로 사용되기 보다는 함수에 포함되어서 함수를 재사용할 목적으로 활용되기 때문에 함수에 대한 이해가 반드시 선행되어야 한다. 함수에 대한 이해가 동반되지 않음으로써 매개변수를 어느 위치에 삽입해야 하는지에 대한 이해도 이루어지지 않는 사례들이 이에 해당된다.

둘째, 함수의 적용 원리에 대한 이해가 부족했다. 함수가 어떤 역할을 하는지에 대한 이해만으로는 프로그래밍을 통해 문제를 해결할 수 없다. 함수를 구성하는 원리, 함수를 호출하는 원리, 함수가 수행되는 절차 등을 파악하고 적용할 수 있어야 실질적인 문제 해결이 가능하다. 학습자들의 응답 중에서 문제 해결 과정에서 함수 블록은 어느 위치에 삽입해야 하는지, 함수 블록 내에는 어떠한 블록들이 포함되어야 하는지에 대해 어려웠다는 반응들은 원리에 대한 이해가 부족한 결과라고 할 수 있다.

셋째, 함수에 포함되어야 하는 패턴을 인식하지 못했다. 이 부분은 반복문에서의 패턴을 인식하는 방법과 비슷하다. 다만 차이점이라면 반복문의 패턴은 문제 해결 과정 한 부분에서 반복되는 단발성을 띄는 것이지만, 함수는 문제 해결 과정 전반에 걸쳐 여러 번 발생하는 패턴을 탐색함으로써 필요할 때 재사용 가능한 것이다. 따라서 보다 넓은 맥락에서 패턴을 찾을 수 있어야 한다. 그러나 학습자들은 문제 해결 과정 전체의 맥락에서 함

수에 들어가야 할 패턴을 인식하지 못하거나, 인식했다고 하더라도 함수 내에 포함되어야 할 블록과 그렇지 않은 블록을 분리하지 못하는 등의 결과를 보였다.

5. 결론

본 연구에서는 프로그래밍 학습 시 초보 학습자가 겪는 어려움을 분석하고 프로그래밍 교육을 위한 교수 학습 방안을 제시하였다. 이를 위하여 초보 학습자의 프로그래밍 교육에 적합한 코드닷오알지를 도입하고, 학습자들이 코드닷오알지의 퍼즐 문제들을 해결하는 동안 이해가 되지 않았던 부분이나 잘못 해결했던 부분, 어려움을 겪었던 이유가 무엇인지를 기록하도록 하였다. 이후 코드닷오알지에 포함된 '순차', '반복', '논리', '이벤트'의 4가지 프로그래밍 개념들을 바탕으로 프로그래밍 개념 학습에서 많은 학습자들이 경험하는 어려움과 그 원인이 무엇인지 분석하였다.

프로그래밍 개념별로 학습자가 경험한 어려움과 그 원인을 정리하면 다음과 같다.

순차 학습자들은 블록의 순서를 제대로 나열하지 못하거나 필요한 블록을 생략 혹은 불필요한 블록을 삽입하는 등의 문제 해결에서의 어려움을 보였다. 이러한 결과는 학습자들이 문제 해결 절차를 체계적으로 작성하지 않고 순간적인 사고에 의존하거나, 큰 문제를 작은 문제로 분해하여 해결하고 합치는 과정에서 필요한 블록을 생략, 현재까지 작성된 프로그램의 결과가 다음 프로그래밍에 영향을 미친다는 사실을 고려하지 않은 데 있으며, 컴퓨터가 문제를 해결하는 관점에서 사고하지 못하고, 필요한 자료 수집이 이루어지지 못했거나 과거 학습 경험의 인지적 조작의 방해를 받은 데에 따른 결과이다.

반복 학습자들은 반복되어야 할 블록 순차적 나열, 반복문 내에 불필요한 블록을 삽입, 반복 횟수 입력 오류, 문제 해결에 적합하지 않은 반복 블록 선택, 중첩 구조를 만들지 못하는 등의 어려움을 느끼고 있었다. 이는 학습자들이 반복 개념에 포함되는 4가지 개념을 정확히 파악하지 못했거나 반복되어야 하는 패턴에 대한 인식 부족, 반복 블록을 사용하는 방법에 대한 인지 불가, 중첩의 구조화에 대한 충분한 파악 부족, 시행착오적 접근 방법에 그 원인이 있다.

조건 학습자들은 조건 개념 내에 반복 기능이 포함되어 있다고 인식하거나 조건 블록 내에 삽입되어야 할 블록을 찾아내지 못하는 어려움을 겪은 것으로 나타났다. 이는 논리 구조와 사용 방법에 대한 인지 부족, 코드닷오알지에서 제시하는 학습 순서에 따른 선행 학습의 영향에 의한 결과이다.

이벤트 학습자들은 함수 대신에 함수에 들어가야 할 블록들을 반복해서 절차적으로 나열하거나 반복 블록을 사용하였으며 함수의 호출 위치 파악이나 매개 변수 삽입 위치 파악, 함수에 포함되는 블록을 선정하는 데에 어려움을 느끼고 있었다. 이는 함수에 대한 개념과 사용 방법에 대한 이해가 충분히 이루어지지 않은 점, 문제 해결 과정의 전체 맥락에서 함수에 포함되어야 하는 패턴을 정확하게 인지하지 못한 데 그 원인이 있다.

이러한 분석 결과를 토대로 프로그래밍 교육을 위한 교수 학습 방안을 제시하면 다음과 같다.

첫째, 프로그래밍 개념을 정확하게 이해하고 내면화하기 위한 전략이 필요하다. 개념에 대한 충분한 이해 없이 해결만을 목적으로 하는 학습은 문제 유형을 기억해두었다가 같은 유형이 등장하면 매치되는 방법을 적용하는 수준에 그치기 때문에 익히지 못한 유형의 문제는 해결하기 어렵다. 또한 유형에 의존적인 이해는 프로그래밍 개념에 대한 오개념이 형성될 가능성을 높이게 되며 원하는 결과 값을 얻기 어렵도록 한다. 따라서 개념에 대한 대략적인 이해 수준을 넘어 각 개념이 문제 해결에 어떤 역할을 하고 왜 필요한지, 어떤 상황에 어떻게 적용될 수 있는지를 정확히 깨닫도록 하는 것이 가장 중요하다.

프로그래밍 개념에 대한 내면화는 학습자가 문제 해결에 적합한 개념을 선택하도록 하며 문제 해결의 효율성 향상을 꾀할 수 있다.

둘째, 문제 해결 절차를 가시화 할 수 있는 정교화된 알고리즘을 작성하는 연습이 충분히 이루어지도록 안내할 필요가 있다. 알고리즘을 머릿속으로 구상하는 수준에서 문제를 해결하는 것은 사람이 갖는 사고의 한계로 인하여 학습자들로 하여금 많은 시행착오를 겪도록 한다. 반면에 정

교하게 짜여진 알고리즘은 문제 해결을 위한 심사숙고의 과정을 거치도록 하기 때문에 계획 없이 작성된 프로그램을 반복적인 시행착오를 통해 해결하려는 소극적인 태도를 개선할 수 있다. 또한 문제 해결 맥락을 논리적이고 체계적으로 살펴볼 수 있도록 하므로 문제 해결에 필요한 블록이 생략되거나 불필요한 블록이 추가되지 않도록 하며, 반복되는 패턴을 인식하거나 중첩 구조를 쉽게 발견하도록 한다. 문제 해결 과정에서의 오류를 탐지하고 수정하는 과정을 용이하도록 한다.

셋째, 문제 해결 과정에 대해 자유롭게 사고할 수 있는 전략을 마련할 필요가 있다. 학습자가 과거에 학습했던 지식이나 문제 해결 경험이 현재 문제 해결에 부정적인 영향을 미치면서 문제 해결의 효율성을 떨어뜨리고 잘못된 해법을 만들어 내는 경우가 있다. 코드닷오알지의 학습 과정 중 일부 퍼즐 문제는 삼각형 한 내각은 60° 라거나 한 번 지나간 자리는 다시 지날 수 없는 규칙을 갖는 한붓그리기의 학습 경험으로 인해 학습자들이 잘못된 해결 절차를 세우거나 문제 해결에 필요한 최소 블록의 수를 초과하도록 하였다.

넷째, 학습해야 하는 프로그래밍 개념을 어떠한 순서로 구성하여 가르칠 것인가에 대한 심도 깊은 설계가 필요하다. 학습 순서는 학습자가 개념을 학습하고 개념들을 구조화함에 있어 학습의 효율성, 오개념 형성에 영향을 미칠 수 있기 때문에 매우 중요하다. 실제로 학습자들은 현재 학습해야 할 프로그래밍의 개념 바로 직전에 배웠던 개념의 영향을 받아 오개념을 형성하거나 시행착오를 겪는 경우가 많다. 코드닷오알지의 논리 개념 학습은 논리 개념 직전에 학습했던 반복의 개념이 개입됨으로써 조건 개념에 반복 수행의 기능이 포함된다고 착각함으로써 논리와 반복 중첩이 필요한 경우 반복문을 생략하는 경우가 있었다.

다섯째, 다양한 문제 해결 경험을 제공할 필요가 있다. 다양한 문제 제시는 학습한 프로그래밍 개념을 내면화하도록 하며 개념이 문제 해결에 어떻게 적용되는지 그 원리를 파악하도록 함으로써 필요한 때에 즉각적으로 쉽게 활용될 수 있도록 한다.

본 연구에서는 초보 학습자들의 프로그래밍 교육을 위해 적합하다고 판단되는 '코드닷오알지'를 기준으로 학습자들이 경험하는 어려움은 무엇인지를 분석하였다 그러나 '코드닷오알지' 외에 다양한 프로그래밍 학습 도구를 도입하고 프로그래밍 학습 도구의 특성에 따라 학습자가 겪는 어려움을 분석함으로써 프로그래밍 학습 효과를 보다 향상시킬 필요가 있다.

참 고 문 헌

[1] International Society for Technology in Education & Computer Science Teachers Association. (2011). *CSTA K-12 Computer Science Standards Revised 2011*. Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf.

[2] Department for Education. (2013). *The national curriculum in England: Framework document. Retrieved from https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/210969/NC_framework_document_-_FINAL.pdf.*

[3] Indian Institute of Technology Bombay. (2013). *CMC: A model computer science curriculum for K-12 schools* 3rd edition(Technical Report: TR-CSE-2013-52).

[4] 배학진・이은경・이영준 (2009). 문제 중심 학습을 적용한 스크래치 프로그래밍 교수 학습 모형. **컴퓨터교육학회논문지, 12**(3), 11-22.

[5] 홍성권・최정원・이영준 (2014). 초등학생의 게임 프로그래밍 경험이 자기효능감에 미치는 영향. **교원교육, 30**(3), 197-215.

[6] 유정수・이민희 (2009). 두리틀을 이용한 프로그래밍 수업이 창의성, 문제해결력, 프로그래밍 흥미도 향상에 미치는 영향. **정보교육학회논문지, 13**(4), 443-450.

[7] 태훈・김종훈 (2013). Kodu를 이용한 프로그래밍 학습이 초등학생의 논리적 사고력과 학습 흥미에 미치는 영향. **컴퓨터교육학회논**

문지, 16(3), 13-22.
[8] 송정범・조성환・이태욱 (2008). 메타인지 전략을 활용한 게임 프로그래밍 학습이 초등학생의 문제해결력에 미치는 효과. **교원교육, 24**(4), 432-447.
[9] 정웅열・이은경・이영준 (2009). 전문계 고등학교 학습자의 동기 유발 및 지속을 위한 로봇 프로그래밍 교수 학습 모형. **컴퓨터교육학회논문지, 12**(4), 13-21.
[10] 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.
[11] Jenkins, T. (2002, 8). On the difficulty of learning to program. *In Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences*. Loughborough.
[12] Milne, I., & Rowe, G. (2002). Difficulties in learning and teaching programming-views of students and tutors. *Educationa and Information Technologies, 7*(1), 55-66.
[13] 임웅 (2011). 창의성, 그 잠재력의 실현을 위하여. **상담과 지도, 46**, 21-32.

최 정 원

2003 충북대학교
컴퓨터과학과(이학사)
2012 한국교원대학교
컴퓨터교육과(교육학석사)

현재 한국교원대학교 컴퓨터교육과 박사과정
관심분야: 정보영재교육, 퍼즐기반학습, 학습과학, 융합교육
E-Mail: cjw0829@daum.net

이 영 준

1988 고려대학교
전산과학과(이학사)
1994 미국 미네소타대학교
(전산학 Ph.D.)

현재 한국교원대학교 컴퓨터교육과 교수
관심분야: 정보통신교육, 지능형시스템, 학습과학
E-Mail: yjlee@knue.ac.kr