초등학생 프로그래밍 언어 학습을 위한 함수 개념 지도 방법 연구

노현정°, 김갑수
서울교육대학교 컴퓨터교육과
zangmi77@hotmail.com, kskim@snue.ac.kr

The Teaching Method of Functions in Programming Language Learning for Elementary Students

Hyeon-Jeong Noh°, Kap-Su Kim
Dept. of Computer Education, Seoul National University of Education

요 약

초등학생 프로그래밍 교육은 프로그래밍 활동을 통해 논리적 사고력과 문제 해결력을 신장시키는 데 의의를 두고 다양한 프로그래밍 교육 방법과 학습 시스템을 개발하려는 연구가 이루어지고 있다. 프로그래밍 교육의 목표가 프로그래밍적 사고력 함양이라면 프로그래밍적 사고를 언어로 표현하여 실제로 프로그램을 작성할 수 있는 프로그래밍 언어 사용 능력 함양도 필요하다. 초등학생 프로그래밍 언어 학습은 특정 언어의 문법적 설명과 해석을 지양하고 프로그래밍 언어에 대한 올바른 개념 이해와 활용을 통해 프로그램을 구현하는데 필요한 기초 소양 능력 함양에 중점을 두어야 한다. 따라서 초등학생을 위한 프로그래밍 언어 교육 방법의 체계화에 기여할 수 있는 하나의 모델로서, 프로그래밍 언어의 기본적인 개념 중 함수 개념을 효과적으로 지도할 수 있는 지도 원리와 학습 모형을 연구하였고, 함수가 가진 특성 즉 함수적 사고 과정을 이용하여 프로그래밍 언어 기술 능력과 논리적 사고력 및 문제해결력의 고등 인지기술 능력을 함께 신장시킬 수 있는 지도 방법을 제안하고자 한다.

1. 서 론

정보화 사회로 접어들면서 학교 컴퓨터 교육의 필요성이 강조되고, 컴퓨터 교육은 단순히 응용 소프트웨어의 활용에 그치는 것이 아니라 컴퓨터를 활용하여 실생활의 여러 문제들을 논리적으로 해결할 수 있는 능력을 함양시킬 수 있어야 한다. 프로그래밍은 컴퓨터를 통해 사고력과 문제해결력을 기를 수 있는 유용한 도구이며 충분한 교육적 가치를 지니고 있다[1].

프로그래밍을 통해 주어진 과제를 처리하기 위해서는 프로그래밍 언어 학습이 이루어져야 하는데, 프로그래밍 언어는 초등학생들이 접근하기에는 매우 어려운 문법적 형식과 구조를 가지고 있기 때문에, 초등학생을 위한 프로그래밍 교육은 프로그래밍 언어 기술 습득에만 치중하지 않는 다양한 접근 방식에 따르며, LOGO와 같은 교육용 프로그래밍 언어를 개발하여 활용하고 있고, 알고리즘 활동, 문제해결 활동 등의 다양한 활동을 통해 프로그래밍적 사고력을 신장시키고자 한다.

그러나 프로그래밍 능력은 실제로 프로그램을 작성하고 구현하여 과제를 처리하는 능력이므로, 초등학생에게도 실제 프로그램을 작성하는데 필요한 기초적인 언어 교육이 필요하다. 초등학생에 있어서 프로그래밍 언어 학습은 언어의 형식과 구조를 완벽히 이해하여 완성된 프로그램을 작성하는 것이 아니라, 기본적인 언어의 개념과 구조를 이해하여 학습자의 사고 과정을 언어로 표현할 수 있는 기초 소양 능력을 갖추는 것이다.

따라서 본 연구는 초등학생을 위한 프로그래밍 언어 교육 방법의 체계화에 기여할 수 있는 하나의 모델로서, 프로그래밍 언어의 기본적인 개념 중 함수 개념을 효과적으로 지도

할 수 있는 방법을 연구하였다. 함수가 가진 특성을 이용하여 프로그래밍 언어기술 능력과 논리적 사고력 및 문제해결력의 고등인지기술 능력을 함께 신장시킬 수 있는 지도 방법을 제안하고자 한다.

2. 이론적 배경 및 관련 연구

2.1 프로그래밍의 교육적 의의[2]

교육적인 측면에서 프로그래밍의 의의는 첫째, 프로그래밍 활동을 통해 학습자 스스로 사고력을 향상시킬 수 있다는 점과 둘째, 새로운 교과내용으로서의 프로그래밍을 이용하여 문제를 풀 수 있는 능력을 갖추는 것은 정보화 사회에 대비하는 지름길이라는 점이다. 여기에서 사고력이란 합리적 사고력, 창의적 사고력, 확산적 사고력 등을 의미하는 포괄적 용어이다.

사고력과 문제해결력을 기르는 것은 학교 교육의 목표에 속하며 이러한 사고력과 문제해결력을 기르는 데 프로그래밍 학습은 유용한 교육적 도구라고 할 수 있다. 즉 프로그래밍 교수-학습은 프로그래밍 언어 습득에만 국한되지 않고 고등인지기술 습득을 위한 훌륭한 방법이 되는데, 프로그래밍 과정에서 요구되는 문제해결력과 지속적 오류 검증 및 수정 작업이 반성적 사고를 신장시킬 수 있다.

2.2 프로그래밍 언어 학습의 필요성[3]

프로그래밍은 컴퓨터를 이용하여 과제를 처리하기 위한 작업 과정을 지시하는 절차적 표현 방법으로서, 프로그래밍 능력은 알고리즘의 구현, 프로그래밍 언어의 활용, 다양한 자원의 효율적 활용 등의 복합적인 능력이 요구되어진다. 그러나 프로그래밍은 인간과 컴퓨터의 매개 언어를 통한 의사소통 과정이므로, 프로그래밍 학습의 초기 단계에는 무엇보다 프로그래밍 언어 학습이 학습자에게 중요한 영역으로 자리하게 되며, 프로그래밍 언어를 효율적으로 지도하는 방법에 대한 연구는 프로그래밍 능력 신장에 중요한 요소로 자리하게 된다.

2.3 프로그래밍 언어 학습의 접근 방법[4]

전통적으로 프로그래밍 학습은 구조론(syntax)을 중심으로 이루어졌기에 프로그래밍의 과정이 컴파일과의 씨름을 통해 이루어졌다. 프로그래밍 교육이 문법에 대한 기계적 암기나 프로그래밍 언어의 사용법을 익히는 데 치중한다면 이는 학습자의 인지 부담이 크며 논리적 사고력을 기르는 데 적합한 방법이 되지 못한다.

최근의 프로그래밍 언어 학습 접근법을 살펴보면, 구조 및 절차학습에서 자유로운 syntax-free approach, 프로그래밍 언어를 읽고 쓸 수 있는 법에 중점을 두는 literacy approach, 문제해결과정을 중심으로 하는 problem-solving approach, 컴퓨터와의 상호작용을 중시하는 computation as interaction 등 여러 가지 접근법들이 대두되고 있다.

2.4 프로그래밍 언어 교수법

1) 제 2언어 교육 관점의 귀납적 교수법

제 2언어 교육에서 학습자의 제 2언어에 대한 문법 지도에 관한 연구는 프로그래밍 언어가 음성 언어가 아닌 규칙 언어라는 점에서 관심을 갖게 한다. 제 2언어의 문법 학습 과정에서 연역법과 귀납법이 자주 사용되며, 연역법은 규칙의 언급에서 구체적인 경우에서의 규칙의 적용으로 이행하는 과정을 의미하며, 귀납법은 구체적인 것에서 일반적인 것으로, 즉 다양한 예들로부터 유도할 수 있는 일반화로 이행하는 과정을 의미한다고 하였다[3].

프로그래밍 언어의 학습도 프로그램 구성의 가장 기본 단위인 명령어의 기능을 학습하고 이를 이용한 단위 프로그램을 학습하는 학습 방법에서 벗어나 다양한 단계별 예문을 중심으로 언어의 규칙을 발견하고 그 개념을 내재

화하여 실제 프로그래밍에 적용하는 일반화 방법으로 개선되어야 할 것이다[5].

귀납적 교수-학습법은 기초적 컴퓨터 기능과 개념 정립이 덜 되어 있는 학습자들에게 기초적 컴퓨터 프로그래밍 능력을 배양하는데 더욱 효과적이라는 연구 결과가 나와 있으며 이는 초보자를 위한 프로그래밍 언어 학습이 어떻게 이루어져야 할 것인가를 시사하고 있다 하겠다[3].

2) 프로그램 시각화을 통한 교수법

프로그램 시각화는 그래픽이 기존의 텍스트 형태로 표현되어진 프로그램의 여러 가지 단면을 시각적 형태로 나타내는 것이다. 주어진 데이터를 가지고 프로그램이 실행될 때 프로그램 변수들의 값과 주요 자료 구조의 형태와 내용을 전체, 혹은 일정 범위에서 수행 과정을 동적으로 표현해주어 프로그램의 구조와 기능을 명확히 이해시킬 수 있게 해준다[6].

같은 내용을 글과 그림으로 함께 제시하여 설명하면 입력된 정보가 제각기 글과 그림으로 표상되면서 장기기억에 이중으로 기호화되므로 효과적인 학습이 일어나게 된다. 즉 학습자는 한 개념에 대하여 그림과 글 두 가지의 단서를 가지게 되고 이를 이용하여 기호화하게 되면 여분의 코딩을 할 가능성이 높게 되며, 그에 따라 기억, 검색, 정보 유출을 용이하게 된다. 따라서 프로그래밍 언어 교육에 있어서 적절한 그림의 사용은 프로그래밍의 개념 정립에 효과적일 것으로 보이며 그에 따라 프로그래밍 능력의 신장을 가져올 것이다[3].

3. 수학적 관련성을 통한 함수 지도 원리

3.1 수학에서의 함수와 함수적 사고

1) 수학에서의 함수 개념

수학에서 함수는 '변수 x의 값의 변화에 따라서 다른 변수 y가 정해질 때 y를 x의 함수'라고 정의한다. 즉, 함수의 개념은 변하는 두 양의 관계를 수학적으로 받아들인 데서 발생하였는데, 예를 들면 서로 관계를 맺고 있는 두 양 A, B가 있을 때, A의 크기가 변함에 따라 B의 크기가 변하게 된다. 이 때 두 양 사이의 관계를 함수라 한다. 그러나 '따라서 변한다'는 의미는 직관적으로 알기 쉽지만 논리적이지 못하다. 이를 논리적으로 설명하기 위해 집합의 관점에서 함수를 다음과 같이 정의한다. 2개의 집합 X, Y에서 X의 각 원소 x가 Y의 단 하나의 원소 y에 대응되는 관계를 'X에서 Y로의 함수'라고 한다. 여기서 대응이란 하나의 값이 정해지면 거기에 따른 다른 값이 정해진다는 생각에서 두 값으로 이루어지는 하나의 쌍이다. 이것이 함수에 대한 근본적인 생각이다[7].

2) 수학에서의 함수적 사고

수학에서 함수적 사고는 '무엇을 정하면 무엇이 정해진다는 점에 주목하거나, 변수 사이의 대응 규칙을 발견 또는 이용하려는 생각'이라고 하고, 다음과 같은 특징을 갖는다. 첫째, 의존 관계에 주목한다는 것이고, 둘째, 대응 규칙을 발견하고 이를 활용하는 것이며, 셋째, 관계의 표현 방법을 탐구하는 것이다[8].

초등학교 수학에서 함수 지도는 함수 그 자체를 가르치는 것이 아니라 함수적인 사고를 배양하고자 한다. '따라서 변하는 변량'에 대한 인식을 가질 수 있게 하고, 2개의 집합 X, Y에서 서로 대응하는 원소를 발견하여 대응의 규칙성을 발견할 수 있게 하며, 대응 관계를 말이나 식으로 표현할 수 있게 하고, 대응관계식을 이용하여 하나의 변량을 알며 다른 변량의 값을 구할 수 있게 하며, 함수적인 관찰에 의해 문제를 해결할 수 있게 한다[7].

3.2 프로그래밍에서의 함수와 함수적 사고

1) 프로그래밍에서의 함수 개념

고급 프로그래밍 언어에서는 프로그램을 모듈화시킬 수 있는 서브프로그램을 지원하고 있다. 서브프로그램은 일반적으로 함수(function)와 서브루틴(subroutine) 두 가지를

제공하는데, 함수와 서브루틴의 차이는 함수가 함수 이름으로 하나의 값을 반환하는데 비하여 서브루틴은 그 이름으로 값을 반환하지 않는데 있다. 서브루틴은 결과 값을 하나 이상의 매개변수에 배정하거나 자신의 환경을 변환하거나, 또는 이 두 가지 일을 다 수행함으로써 주어진 목적을 완성하는 프로시저이고, 함수는 하나의 결과 값만을 반환하는 프로시저로서 식에서 변수처럼 하나의 원소로 사용할 수 있다[9].

즉 함수는 인자(argument, or parameter) 값을 입력받아 일련의 실행을 마친 뒤 하나의 반환(return) 값을 출력하는 서브프로그램의 일종으로 이는 수학적인 함수 개념과 일치한다. 수학에서 '변수 x의 값의 변화에 따라서 다른 변수 y가 정해질 때 y를 x의 함수'라 할 때, 변수 x의 값은 함수 호출시 함수로 입력되는 인자 값이 될 수 있고, 변수 y의 값은 함수 실행 후의 결과가 출력되는 반환 값이 될 수 있다.

2) 프로그래밍에서의 함수적 사고

서브프로그램은 일련의 실행을 단순화하기 위해서 프로그래밍 언어에서 제공하는 메커니즘으로, 프로세스는 호출하는 서브프로그램의 이름과 인자를 명시함으로써 원하는 일련의 실행을 행할 수 있다. 서브프로그램을 사용하는 이유는 자주 사용되는 코드를 반복하는 것보다 서브프로그램으로 묶어 필요할 때마다 호출하는 것이 경제적이기 때문이다. 이는 대형 프로그램을 작성하기 위해 프로그램을 보다 작고 이해할 수 있는 단위로 나누어야 한다는 프로그래밍의 설계 원칙에 따른 것으로 '모듈화 프로그래밍(modular programming)'이라 한다[10].

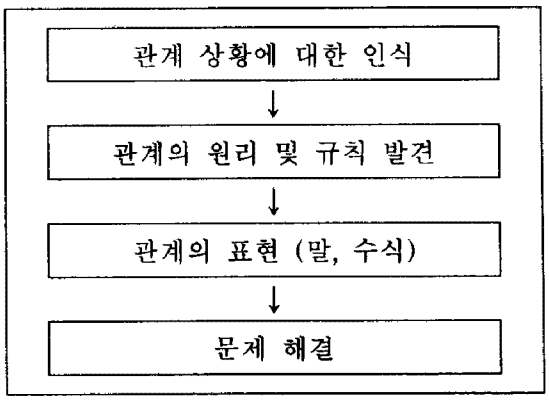
이러한 관점에서 프로그래밍에서의 함수적 사고는 프로그래밍을 통한 문제해결과정에서 문제를 분석하고, 모듈화할 수 있는 프로그램 동작 단위를 찾아내어, 프로그램의 입출력 값과 프로그램 내의 동작을 함수로 블록화하여 표현하기까지 일련의 사고 과정이라 할 수 있다. 나아가 함수 호출에서 호출한 프로그램과 호출된 서브프로그램 간에 값이 어떻게 전달되는지를 이해하고 관계성을 표현하는 사고 과정도 포함한다.

초등학생의 프로그래밍 언어 교육에서 함수 지도는 개별 언어의 문법적 규칙에 따라 함수를 정의하고 호출하는 방법을 가르치는 것이 아니라, 함수 개념을 바르게 이해하고 도입하여 문제를 해결할 수 있는 함수적 사고를 배양하도록 하는 것이 보다 바람직하다. 이는 초등학생 수준의 프로그래밍 언어 교육이 단순한 언어 기술 습득이 아닌 인지적 사고력과 문제해결력 등의 고등인지기술 습득을 지향한다는 점에도 잘 부합된다.

3.3 함수적 사고를 통한 문제해결과정

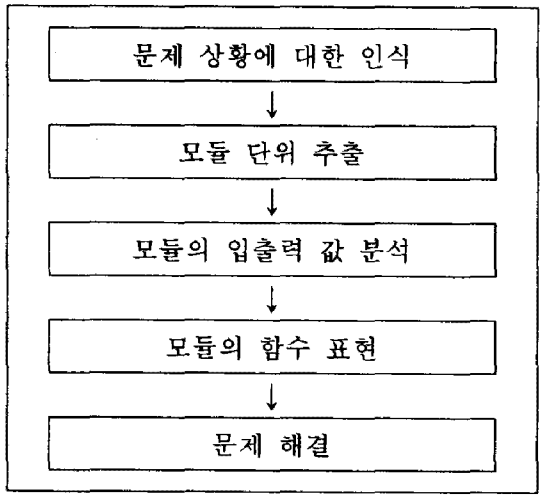
프로그래밍에서의 함수의 개념이 수학에서의 함수의 개념에서 도출되었다면, 프로그래밍에서의 함수적 사고는 수학에서의 함수적 사고와 밀접한 관련이 있다. 이러한 관련성을 발전시켜 '수학에서의 함수적 사고를 통한 문제해결과정'을 토대로 '프로그래밍에서의 함수적 사고를 통한 문제해결과정'을 정리하고, 나아가 두 사고과정이 통합된 '통합적인 함수적 사고를 통한 문제해결과정'을 새롭게 정립하고자 한다.

수학에서의 함수적 사고를 통한 문제해결과정은 <그림 1>과 같다.



<그림 1> 수학에서의 함수적 사고를 통한 문제해결과정

이를 토대로 프로그래밍에서의 함수적 사고를 통한 문제해결과정을 <그림 2>와 같이 정리하였다.

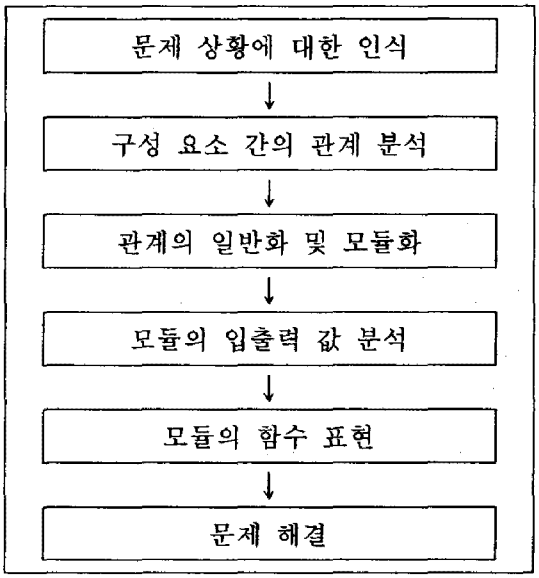


<그림 2> 프로그래밍에서의 함수적 사고를 통한 문제해결과정

수학에서의 함수적 사고는 수학적 상황에서 문제를 구성하고 있는 요소들 간의 관계성을 인식하고 관계의 원리 및 규칙을 발견하여 말이나 수식을 통해 표현함으로써 문제해결점에 도달한다. 프로그래밍에서의 함수적 사고는 여러 가지 문제 상황을 인식하고 분석하여 모듈화할 수 있는 프로그램 동작 단위를 찾아내고, 프로그램의 입출력 값과 프로그램 내의 동작을 함수로 표현함으로써 문제해결점에 도달한다. 프로그래밍에서 주어지는 문제 상황은 반드시 수학적 상황이 아니며 구현된 함수의 내부 과정은 반드시 수학적 논리를 포함하지 않는다.

그러나 학습자에게 주어지는 프로그래밍 문제 상황이 수학적 상황이고, 문제를 해결하는 과정에서 수학적인 함수적 사고가 필요하다면 두 사고과정은 통합될 수 있다. 문제를 이루고 있는 구성요소 간의 관계성을 말이나 수식으로 표현하는 대신 프로그래밍 언어로 표현하는 것이다. 그리고 하나의 문제 상황에는 여러 가지 관계성이 포함될 수 있으므로 이것을 여러 개의 모듈로 구조화할 수 있다.

이에 통합적인 함수적 사고를 통한 문제해결과정을 <그림 3>과 같이 정리하였다.



<그림 3> 통합적인 함수적 사고를 통한 문제해결과정

3.4 수학적인 문제에서 프로그래밍적인 문제로의 전환

앞에서 살펴본 바와 같이 프로그래밍 언어 학습에서 함수 개념은 함수적 사고와 떨어져 학습될 수 없고, 함수적 사고는 수학적인 함수적 사고와의 연관성 및 확장성을 고려할 때 효과적으로 학습될 수 있다. 그 연관성과 확장성은 초등학교 수학과 교육과정에서 함수적 사고를 통해 해결할 수 있는 문제를 추출하여 프로그래밍 학습의 문제로 전환하는 방법으로 실현될 수 있다.

초등학교 수학에서 함수적 사고를 요하는 문제는 대부분 '규칙성과 함수'영역에 속해 있다[7]. '규칙성과 함수'영역의 학습 내용에는 규칙성과 패턴, 함수, 비와 비례 등이 포함되어 있는데 여기서 필요한 학습 내용은 함수 영역에 속하는 문제들이다. 그 중에서 프로그래밍 언어로 표현하여 처리 가능한 문제들이며 이 과정에서 반드시 함수가 활용될 수 있고 함수적 사고 과정이 포함될 수 있는 문제여야 하겠다.

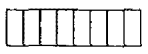
실제 수학과 '규칙성과 함수' 영역에서 수학적 함수 문제 사례를 추출하여 프로그래밍적 함수 문제로 전환하는데 바탕이 되는 함수

적 사고를 분석하였고, 이는 <표 1>과 같다.

<표 1> '규칙성과 함수' 영역의 수학적 함수 문제와 프로그래밍적 함수 사고의 예

<table>
<tr><th>단계
주제</th><th>수학적 함수
문제</th><th>프로그래밍적 함수
사고 과정</th></tr>
<tr><td rowspan="6">4-가

문제를 간단히 하여 해결하기</td><td rowspan="3">4학년 1반은 정사각형 모양의 알림판 가장자리를 꽃그림으로 꾸미려고 합니다. 오른쪽 그림과 같이 한 변에 20장씩 붙인다면, 꽃 그림은 모두 몇 장이 필요한지 알아보시오.</td><td>정사각형 변의 수와 한 변의 꽃그림 수의 관계 분석</td></tr>
<tr><td>'변의 수×물체 수-변의 수' 수식관계 일반화 및 모듈화</td></tr>
<tr><td><입력 값> 변의 수(x1), 한 변의 물체 개수(x2)
<출력 값> 물체 총 개수(y)</td></tr>
<tr><td rowspan="3">오늘은 월요일이고 1일입니다. 오늘부터 100일 후가 운동회날이라면, 운동회날은 무슨 요일인지 알아보시오.</td><td><함수 표현> x1와 x2가 0일 경우 출력 값도 0 조건 표현
int func(int x1, int x2) {
int y;
if ((x1==0)||(x2==0))
return 0;
else { y=x1*x2-x1;
return y;
}
}</td></tr>
<tr><td>요일과 며칠 후의 관계 분석</td></tr>
<tr><td>'며칠 후를 7일로 나눈 나머지 날짜만큼 오늘 요일부터 변한다' 말로 표현한 관계의 일반화 및 모듈화</td></tr>
<tr><td rowspan="6">4-가

규칙을 찾아서 수로 나타내기</td><td rowspan="6">다음 그림에는 직사각형이 모두 몇 개 있는지 알아보시오.
  </td><td>단위직사각형의 개수와 전체 직사각형의 개수 관계 분석</td></tr>
<tr><td>'전체직사각형 개수는 1부터 단위직사각형 개수까지의 합이다' 말로 표현한 관계의 일반화 및 모듈화</td></tr>
<tr><td><입력값>
단위 직사각형의 개수(x)
<출력값>
전체 직사각형의 개수(y)</td></tr>
<tr><td><함수표현> 1부터 x까지의 합을 for 반복문 표현
int func(int x) {
int y=0; int i;
for(i=1 ; i<=x ; i++)
y=y+i;
return y;
}</td></tr>
<tr><td rowspan="5">4-나

두 수 사이의 관계를 알아보기</td><td rowspan="5">미술 시간에 색 테이프를 가위로 잘랐습니다. 한 번 자르면 색 테이프는 2도막이 됩니다. 한 번씩 자를 때마다 색 테이프는 몇 도막이 되는지 알아보시오.</td><td>색테이프 자른 횟수와 도막의 수의 관계 분석</td></tr>
<tr><td>'도막 수 = 자른 횟수 + 1' 수식관계 일반화 및 모듈화</td></tr>
<tr><td><입력 값> 자른 횟수(x)
<출력 값> 도막의 수(y)</td></tr>
<tr><td><함수 표현>
int func(int x) {
int y=0;
y=x+1;
return y;
}</td></tr>
<tr><td></td></tr>
<tr><td rowspan="5">4-나

두 수 사이의 관계를 알아보기</td><td rowspan="5">대응표를 보고 □와 △의 관계를 말하여 보시오.
<table><tr><td>□</td><td>11</td><td>12</td></tr><tr><td>△</td><td>6</td><td>7</td></tr></table></td><td>□와 △의 관계 분석</td></tr>
<tr><td>'△ = □ - 5' 수식관계 일반화 및 모듈화</td></tr>
<tr><td><입력 값> □(x)
<출력 값> △(y)</td></tr>
<tr><td><함수 표현>
int func(int x) {
int y=0;
y=x-5;
return y;
}</td></tr>
<tr><td></td></tr>
<tr><td rowspan="5">5-가

여러 가지 방법으로 문제 풀기</td><td rowspan="5">성냥개비를 늘어놓아 정삼각형 17개를 만들려고 합니다. 성냥개비는 모두 몇 개가 필요합니까?</td><td>정삼각형 개수와 필요한 성냥개비 개수의 관계 분석</td></tr>
<tr><td>'성냥개비수=정삼각형×2+1' 수식관계 일반화 및 모듈화</td></tr>
<tr><td><입력 값> 정삼각형 수(x)
<출력 값> 성냥개비 수(y)</td></tr>
<tr><td><함수표현>
int func(int x) {
int y=0;
y=x×2+1;
return y;
}</td></tr>
<tr><td></td></tr>
<tr><td rowspan="5">5-가

여러 가지 방법으로 문제 풀기</td><td rowspan="5">우리 반 학생 29명이 오목대회를 하는데 모두 서로 한 번의 경기를 해야 합니다. 치러지는 경기는 모두 몇 번일까요?</td><td>학생의 수와 치러지는 경기의 수의 관계 분석</td></tr>
<tr><td>'경기의 수는 1부터 (학생 수-1)까지의 합이다' 말로 표현한 관계의 일반화 및 모듈화</td></tr>
<tr><td><입력값> 학생의 수(x)
<출력값> 경기의 수(y)</td></tr>
<tr><td><함수표현> 1부터 x-1까지의 합을 for 반복문 표현
int func(int x) {
int y=0; int i;
for(i=1 ; i<x ; i++)
y=y+i;
return y;
}</td></tr>
<tr><td></td></tr>
</table>

4. 시각화를 통한 함수 지도 원리

4.1 함수 지도에서 시각화의 필요성

초등학생을 위한 프로그래밍 언어 학습은 단순히 개별 언어 특성에 따른 문법과 규칙을 학습하는 것을 지양하고, 실제로 초등학생 수준에서 영어로 표기된 프로그래밍 언어를 학습하여 프로그램 코드를 작성한다는 것은 대단히 어려운 일이다. 따라서 함수 표현에 있어서 시각화 형식을 정의하여 활용함으로써 함수에 대한 개념을 보다 확실하게 이해시킬 수 있고 초등학생들의 사고를 보다 쉽게 표현할 수 있도록 도와줄 수 있다.

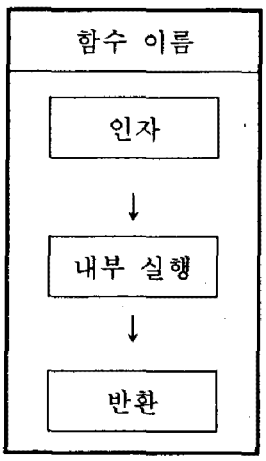
시각화 단계는 프로그래밍 언어 기술의 선행 단계가 되어야 하고 시각화 형식은 프로그램 코드의 일반화된 모델로서 프로그램 코드로의 전환이 가능해야 한다. 초등학생에게 함수를 지도할 때 필요한 시각화 형식은 함수 구성 부분과 함수 호출 부분으로 구체적인 내용은 다음과 같다.

4.2 시각화 형식의 정의

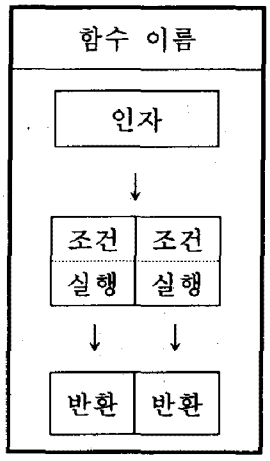
1) 함수 구성의 시각화 형식

함수 구성의 시각화는 하나의 함수로 표현되는 하나의 블록 영역을 함수 이름, 인자(입력), 내부 실행, 반환(출력)의 네 가지 영역으로 나누어 각각의 영역에 알맞은 함수 구성 요소를 표현하는 것이다. 함수 이름 영역에는 함수 호출시 명시되는 이름만을, 인자와 반환 영역에는 유형을 배제한 변수 이름만을 표현한다. 내부 실행 영역에는 개별 언어의 문법을 따르지 않고 인자 값을 입력받아 반환 값을 내기까지의 실행 과정을 말이나 수식을 통해 표현한다. 관계는 화살표(→)를 통해 표현하는데, 내부 실행 과정에서 조건 분기에 따라 반환 값이 달라진다면 내부 실행 영역을 조건에 따라 구분하고 그에 따른 반환 값 영역을 구분하여 표현한다.

함수 구성의 시각화 형식은 <그림 4>와 <그림 5>와 같다.



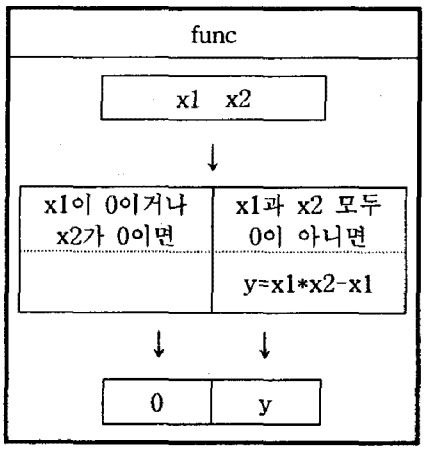
<그림 4> 함수 구성의 시각화 형식

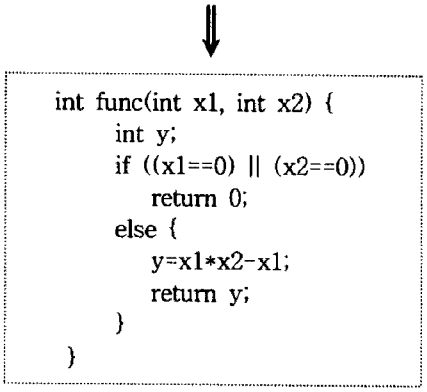


<그림 5> 조건 분기 포함한 함수 구성의 시각화 형식

실제 문제를 해결하는 과정에서 함수 구성의 시각화 형식을 사용한 예는 <그림 6>과 같다.

4학년 1반은 정사각형 모양의 알림판 가장자리를 꽃 그림으로 꾸미려고 합니다. 오른쪽 그림과 같이 한 변에 20장씩 붙인다면, 꽃 그림은 모두 몇 장이 필요한지 알아보시오.	정사각형 변의 수와 한 변의 꽃그림 수의 관계 분석
	'변의 수×물체 수-변의 수' 수식관계 일반화 및 모듈화
	<입력 값> 변의 수(x1), 한 변의 물체 개수(x2) <출력 값> 물체 총 개수(y)
	<함수 표현> x1와 x2가 0일 경우 출력 값도 0 조건 표현



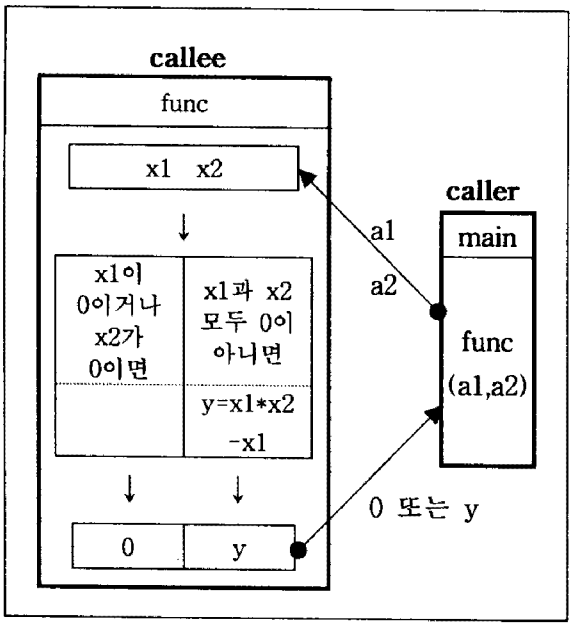


<그림 6> 함수 구성의 시각화 형식 사용 예

2) 함수 호출의 시각화 형식

함수 호출의 시각화 형식은 이미 구성된 함수를 호출하는 방법의 시각화로, 함수를 호출할 때 전달하는 인자 값과 실행 후 함수가 반환하는 결과 값을 표현한다. 값의 전달이 일어나는 시점과 방향은 화살표(●→)로 나타낸다.

함수 호출의 시각화 형식은 앞에서 살펴본 예를 들어 <그림 7>과 같이 나타낼 수 있다.



<그림 7> 함수 호출의 시각화 형식 사용 예

4.3 시각화 형식의 효과

함수 지도에서 시각화 형식을 정의하여 활용하였을 때 다음과 같은 효과가 기대된다.

첫째, 함수 구성의 시각화 형식은 함수 구조를 시각적으로 보여주기 때문에 함수 개념을 직관적으로 이해하는 데 도움이 된다. 하나의 단위 프로그램으로서 입력, 실행, 출력 과정을 가지는 추상적인 함수 개념을 시각적으로 표현해봄으로써 개념 이해를 촉진시킬 수 있다.

둘째, 함수 호출의 시각화 형식은 함수 호출방법을 시각적으로 보여주기 때문에 실제 프로그램 작성 시 필요한 함수 활용 능력을 향상시키는 데 도움이 된다.

5. 함수 지도 중심 프로그래밍 언어 학습 모형

앞에서 연구한 '수학적 관련성을 통한 함수 개념 지도 원리', '시각화를 통한 함수 개념 지도 원리'를 토대로 '함수 지도 중심 프로그래밍 학습 모형'을 제안하고 한다.

이 모형의 목적은 초등학생들의 함수 개념 이해를 통한 프로그래밍 능력 향상과 함수적 사고를 통한 문제해결능력 향상이다. 이 모형의 의의는 함수의 특성을 이용하여 프로그래밍 언어기술과 고등인지기술을 동시에 학습할 수 있는 방법을 제시함으로써 초등학생을 위한 프로그래밍 언어 학습 방법의 체계화에 기여하는 데 있다.

이 모형의 제한점은 다음과 같다.

첫째, 프로그래밍 언어에서 여러 가지 기본적인 개념을 배제하고 함수 개념만을 다루었다. 따라서 이 모형은 변수와 제어구조의 기본적인 개념 이해와 활용에 대한 선행학습이 이루어진 학습자에게 함수를 지도할 때 적용할 수 있다.

둘째, 함수 개념 중에서 함수 호출을 통한 호출자와 함수의 다양한 관계, 호출자와 함수 간의 값 전달 방식, 여러 개의 함수 간의 관계는 배제하고 하나의 함수 구현을 통한 문제해결 과정만을 다루었다. 초등학생 수준에서 복잡한 함수의 관계를 이해하고 프로그래밍 언

어로 표현한다는 것은 매우 어렵고 추상적인 과정이기 때문이다.

함수 지도 중심 프로그래밍 언어 학습 모형은 <표 2>와 같이 4단계로 이루어진다.

<표 2> 초등학생을 위한 함수 지도 중심 프로그래밍 언어 학습 모형

문제 인식 탐색	수학 함수 개념 및 함수적 사고 문제 상황 인식 및 해결 방법 탐색 관계의 발견 및 관계의 일반화
함수 동작 분석	프로그래밍 함수 개념 및 함수적 사고 관계의 모듈화 입력, 실행, 출력의 함수 동작 분석 인자(입력) 값과 반환(출력) 값 추출 함수 내부 기능 구체화
시각화 표현	시각화 형식에 의한 함수 표현 인자 값과 반환 값의 변수 표현 함수 내부 실행의 형식화된 표현 학습자의 수준에 따라 학습 종료 가능
언어 표현	언어 형식에 의한 함수 표현 언어 문법과 규칙에 따라 함수 작성 함수 호출에 의한 함수 사용 프로그램 완성 및 오류 수정

5.1 문제의 인식 및 탐색 단계

처음 문제 상황을 접하여 문제를 인식하고 해결 방법을 탐색하는 단계이다. 여기서 문제는 수학에서의 함수적 사고로 해결할 수 있는 문제로 초등학교 수학과 '규칙성과 함수' 영역에서 추출한다. 해결 방법 탐색 과정에서 작용하는 수학적 함수 개념 및 함수적 사고는 구성요소 간의 관계를 발견하고 관계를 일반화할 수 있는 규칙을 찾는 것이다.

<사례> '오늘은 월요일이고 1일입니다. 오늘부터 100일 후가 운동회하는 날이라면 운동회하는 날은 무슨 요일인지 알아봅시다.'라는 문제가 있다. 문제에서 구하려고 하는 것이 '며칠 후의 요일'이라는 점을 인식하고, 함수적 사고를 통해 문제를 탐색할 수 있도록 유도하여 이 문제가 '요일'과 '며칠 후'의 관계를 통해 해결할 수 있다는 것을 발견하도록 한다.

5.2 함수의 동작 분석 단계

문제 인식 및 탐색 단계에서 작용했던 수학적인 함수적 접근이 프로그래밍적인 함수적 접근으로 전환됨으로써 모듈 단위로 함수 동작을 분석하는 단계이다. 변수들의 관계가 말이나 수식으로 표현되던 함수적인 생각이 프로그래밍적인 입력, 실행, 출력 동작으로 전환되는 것이다. 인자(입력) 값과 반환(출력) 값을 추출하고 함수 내부에서 실행되어야 기능을 구체화한다.

<사례> '요일'과 '며칠 후'의 관계성을 하나의 함수로 모듈화 해야겠다는 생각을 하도록 한다. 100일 후 뿐 아니라 200일 후, 300일 후의 요일도 구할 수 있는 함수를 구현하기 위해 인자 값을 '며칠 후'와 '오늘의 요일', 반환 값을 '며칠 후의 요일'로 추출한다. 그리고 '며칠 후를 7일로 나눈 나머지 날짜만큼 오늘 요일부터 변한다'로 일반화했던 관계성을 함수 내부에 구체적으로 어떻게 표현할 것인가를 고민하도록 한다.

5.3 함수의 시각화 표현 단계

함수 동작 분석 단계에서 분석된 내용이 시각화 형식에 의해 함수로 표현되는 단계이다. 추출한 인자 값과 반환 값을 변수로 지정하고, 함수 내부 실행은 변수를 사용하여 형식화된 말이나 식으로 표현한다. 이는 언어의 문법적 규칙에 구애받지 않고 프로그래밍 능력을 신장시킬 수 있는 방법이고 초등학생 수준에서는 실제적인 학습 종료 단계라 할 수 있다.

<사례> 인자 값 '며칠 후'를 변수 'x', '오늘의 요일'을 변수 'd1', 반환 값 '며칠 후의 요일'을 변수 'd2'로 지정하고 'd2=(d1+x%7)%7'라는 함수 내부 실행을 시각화 형식으로 표현하도록 한다.

5.4 함수의 언어 표현 단계

시각화로 표현하였던 함수를 실제 프로그래

밍 언어로 표현하는 단계이다. 기본적인 프로그램 작성과 실행이 가능한 학습자에게 해당되는 단계로, 표현 가능한 언어의 문법과 규칙에 맞게 함수를 표현하고 함수 호출을 통해 프로그램 내에서 함수를 사용하는 방법에 대해 학습한다. 즉, 표현 가능한 프로그래밍 언어를 선택하여 프로그램을 작성하여 실행시켜 보고 오류 수정 활동을 통해 완전한 문제해결점에 도달한다. 바로 이 단계는 초등학생 수준에서 프로그래밍 언어 표현을 쉽게 도와주는 학습 시스템 개발의 필요성을 시사하며 이는 본 연구의 최대 향후 과제라 하겠다.

6. 결론 및 과제

초등학생을 위한 프로그래밍 언어 학습은 기초적인 프로그래밍 언어 사용 능력과 논리적 사고력 및 문제해결력의 고등인지기술 능력을 함께 향상시킬 수 있는 방향으로 나아가야 한다. 이를 위해 본 연구는 함수를 중심으로 수학적 관련성을 통한 함수 개념 지도 원리와 시각화 형식을 통한 함수 개념 지도 원리를 새롭게 정립하고 함수 지도 중심 프로그래밍 언어 학습 모형을 제시하였다.

함수 개념 지도 방법을 중점적으로 연구한 까닭은 함수가 가진 특성 즉 함수적 사고 과정이 프로그래밍 언어가 가진 다른 개념보다 학습자의 사고력을 신장시키는 데 많은 도움을 주고, 문제해결과정으로서 프로그래밍 능력을 향상시키는 데 중요한 요소가 되기 때문이다. 실제 본 연구에서 제안한 지도 원리와 학습 모형은 함수 개념 이해와 활용을 바탕으로 한 프로그래밍 언어 사용 능력 향상과 함수적 사고 과정을 바탕으로 한 문제해결능력 신장에 많은 도움을 줄 거라 기대한다.

무엇보다 본 연구는 단순히 함수 개념을 효과적으로 지도하기 위한 방법 연구가 아니고 초등학생 프로그래밍 언어 교육을 위한 교수-학습 방법 연구에 있어서 본 연구가 하나의 모델로서 기여하고자 함이다.

이에 본 연구가 가지는 향후 과제는 다음과 같다. 첫째, 실제 초등학생을 대상으로 지도 원리와 학습 모형에 따라 함수 개념과 함수적 사고를 지도한 후 그 효과를 검증하는 것이고, 둘째, 초등학생들이 함수적 문제를 프로그래밍으로 해결하는데 있어 보다 쉽게 프로그래밍 언어로 표현하고 코딩하여 실행할 수 있는 그리고 스스로 오류 수정 단계를 경험할 수 있는 프로그래밍 언어 학습 시스템을 개발하는 것이다.

7. 참고문헌

[1] 박성진, "웹 기반 베이식 프로그래밍 튜터 시스템", 대구교육대학교 교육대학원 석사학위논문, 2002.

[2] 이성근, "문제해결력 향상을 위한 웹 기반 프로그래밍 학습 시스템", 대구교육대학교 교육대학원 석사학위논문, 2003.

[3] 안신희, "효과적인 프로그래밍 언어 교육에 관한 연구", 고려대학교 교육대학원 석사학위논문, 2002.

[4] 이경화, "초등학생을 위한 로고 프로그래밍 지도 방안", 한국정보교육학회 2002년 하계 학술발표논문집, 제7권, 제2호, pp.303-310.

[5] 김승주, "순서도 작성을 기반으로 한 C 프로그래밍 언어 학습 시스템", 고려대학교 교육대학원 석사학위논문, 2002.

[6] 임명선, "프로그래밍 교육을 위한 시각화 도구 설계 및 구현", 고려대학교 교육대학원 석사학위논문, 2003.

[7] 오순임, "'규칙성과 함수' 영역의 교수 · 학습 방법 탐색", 부산교육대학교 교육대학원 석사학위논문, 2002.

[8] 최성필, "타 교과와 연결된 상황 설정을 통한 함수적 사고 지도 방안", 경인교육대학교 교육대학원 석사학위논문, 2003.

[9] 원요현, "프로그래밍 언어론", 정익사, 2004.

[10] 김일민, 조세홍, "프로그래밍 언어론", 21세기사, 2003.