

문제해결력 향상을 위한 과제 중심 스크래치 프로그래밍 학습 프로그램

이은경[†]

요 약

프로그래밍 학습은 현대사회가 요구하는 복잡한 문제해결력 향상에 도움이 되지만, 학습자에게 과도한 인지적 부담을 줄 수 있다. 따라서 인지적 부담을 감소시키기 위한 프로그래밍 도구의 선택 및 학습 효과를 최대화하기 교수 학습 전략이 요구된다. 본 연구에서는 중학생의 문제해결력 향상을 위한 과제 중심 스크래치 프로그래밍 학습 프로그램을 개발하고 해당 프로그램이 학습자의 문제해결력에 미치는 영향을 분석하였다. 연구결과, 과제 중심 스크래치 프로그래밍 학습을 수행한 집단이 일반적인 스크래치 프로그래밍 학습과 비주얼 베이식 프로그래밍 학습을 실시한 집단에 비해 문제해결력이 유의하게 향상되었다. 특히 오류를 발견하고 수정하는 능력인 문제점 해결 영역에서 유의한 차이를 더 크게 드러냈다.

주제어 : 프로그래밍 학습, 교육용 프로그래밍 언어, 스크래치, 과제 중심 학습

A Task Centered Scratch Programming Learning Program for Enhancing Learners' Problem Solving Abilities

EunKyoung Lee[†]

ABSTRACT

Programming learning may help to enhance learners' complex problem solving abilities. However, it may cause excessive cognitive loads for learners. Therefore, selection of programming tools and design of teaching and learning strategies to minimize the learners' cognitive loads and to maximize the learning effects. A task centered Scratch programming learning program was developed to enhance problem solving abilities of middle school students. And then, we implemented the developed program in middle school programming classes and analysed the educational effects of the developed program. We found that the developed program was helpful in enhancing learners' problem solving abilities, especially in the element of 'troubleshooting', which explains ability of error detecting and correcting.

Keywords : Programming Learning, Educational Programming Language, Scratch,
Task Centered Learning

[†] 중신회원: 경기 청운중학교 교사
논문접수: 2009년 9월 28일, 심사완료: 2009년 11월 20일

1. 서 론

문제해결력은 현대사회가 요구하는 핵심 역량으로 실생활의 문제해결 뿐 아니라 모든 학문 분야의 전문성 증진의 기초가 되는 인지 기술이다. 따라서 실세계의 복잡한 문제들을 다루기 위한 인지 능력 향상을 위한 교육 방안의 탐색은 현대사회가 요구하는 교육 목표에 가장 부합하는 일이라고 할 수 있다. 최근까지 이러한 노력들을 통해 문제해결력 향상을 위한 다양한 교수 학습 방법들과 이론적 교수 학습 설계 모형들이 제시되어왔다. 그러나 '어떻게 가르칠 것인가'에 관한 교수 방법을 논의하기에 앞서 '무엇을 가르칠 것인가'를 면밀히 고려할 필요가 있다. 이는 향상된 교수 학습 방법이 학습자의 학습 증진을 이끌 수 있지만, 새로운 교수 학습 방법을 적용하기 전에 새로운 교수 학습 내용의 전환을 통해 보다 나은 향상과 근본적인 문제를 해결할 수 있기 때문이다.

컴퓨터 프로그래밍의 과정은 문제해결을 위한 추상적 사고 과정을 컴퓨팅 장치를 통해 모델링하거나 자동화하기 위한 과정이며, 이러한 과정을 경험함으로써 추상적 사고를 강화할 수 있다[1][2][3]. 즉, 프로그래밍의 과정은 문제해결을 위해 적합한 추상적 개념을 선택하고 구성하기 위한 추상화 능력과 추상적 개념을 자동화하기 위해 적합한 컴퓨팅 도구를 선택하고 사용하기 위한 자동화 능력을 요구한다[3]. 이러한 인지적 능력이 다른 학문 분야에서 다루는 문제해결기술이나 추론기술과 다른 점은 보다 일반적인 문제해결상황에 적용할 수 있는 전략적 지식을 생성하고, 컴퓨팅 장치를 통해 문제해결과정을 실제로 구현하고 자동화한다는 측면에서 가장 정확하고 명확한 해법을 제시할 수 있다는 점이다[4].

따라서 실세계의 복잡한 문제해결력 향상을 위해 프로그래밍 교육이 필요하며, 이를 효과적으로 지원하기 위한 교수 학습 도구 및 내용의 선택, 그에 따른 효과적인 교수 학습 방법에 관한 연구가 필요하다. 이에 따라 본 연구에서는 문제해결력 향상을 위한 프로그래밍 학습을 지원하기 위한 프로그래밍 도구로 교육용 프로그래밍 언어인

스크래치를 선택하고, 해당 도구의 효과성을 최대화하고 프로그래밍 학습 과정 중 실세계와 유사한 문제해결경험을 제공하기 위한 과제 중심 교수 학습 전략을 개발하였다.

2. 관련 연구

2.1 교육용 프로그래밍 언어

교육용 프로그래밍 언어는 프로그래밍 교육 뿐 아니라 수학이나 과학, 음악, 미술 분야 등 다양한 학문 분야에서 널리 활용되고 있는 교수 학습 도구이다. 최근 개발된 교육용 프로그래밍 언어들은 단순한 프로그래밍 도구의 차원을 넘어 풍부한 멀티미디어 학습 환경을 동시에 제공한다. 멀티미디어 환경을 제공하는 대표적인 교육용 프로그래밍 언어는 스퀸 이토이(Squeak Etoys), 앨리스(Alice), 스크래치(Scratch) 등이 있으며, 이러한 언어들을 활용한 연구 결과들은 어린 학습자의 동기 유발 및 문제해결력 향상에 긍정적인 영향을 주었음을 보여주고 있다[5][6].

2.2 스크래치

스크래치는 만 8세에서 16세 사이의 어린 학생들의 Computational Thinking 능력 향상을 목표로 개발된 교육용 프로그래밍 언어이다. 따라서 컴퓨터 및 프로그래밍 관련 기술의 향상과 함께 새로운 IT 기술을 스스로 학습할 수 있는 능력, 이를 활용하여 창의적으로 문제를 해결하거나 프로젝트를 설계할 수 있는 능력 향상 지원을 위해 설계되었다. 이러한 스크래치의 특성 및 관련 연구 결과들은 어린 학습자의 문제해결력 향상을 위한 스크래치 활용 교육의 가치를 보여주고 있으며 구체적으로 살펴보면 다음과 같다.

첫째, 도구 사용으로 인한 학습 외적 인지 부하를 감소시킬 수 있다. 스크래치는 직관적인 인터페이스와 함께 전 세계 다양한 국가의 언어를 지원한다. 따라서 도구 자체 사용의 어려움이나 외국어로 구성된 프로그래밍 언어를 이해하는데 소모되는 인지부하를 감소시킬 수 있다.

둘째, 실생활의 문제해결과정과 유사한 경험을

제공할 수 있다. 스크래치는 시각적이고 상호작용적인 프로그래밍 환경을 제공한다. 다양한 멀티미디어 저작 환경을 포함하고 있기 때문에 실세계와 유사한 문제 상황과 조건에 따른 처리 과정을 시뮬레이션 할 수 있다.

셋째, 즉각적이고 구체적인 피드백을 제공한다. 스크래치는 프로그램 블록들을 조합하여 스크립트를 작성하고 작성된 프로그램의 실행결과를 무대 위에서 즉각적으로 확인할 수 있다. 이러한 환경은 어떤 현상에 대한 문제를 발견하고 오류를 수정하는 사고 능력 향상에 긍정적으로 작용한다.

2.3 복잡한 문제와 과제 중심 학습

현대 사회는 실세계의 복잡한 문제해결을 위한 인지적 능력을 요구한다. 그러나 실세계의 많은 문제들은 불명확하고 복잡하며 역동적인 속성을 가지고 있으며, 이러한 속성들은 문제해결에 많은 어려움을 야기한다[7][8].

따라서 문제해결교육은 과거의 각 학문 분야별로 분리되고 단편화된 전문성 신장을 위한 교육 내용 및 방법에서 벗어나 다양한 문제해결상황에 일반적으로 적용할 수 있는 전략적 지식의 습득과 문제의 복잡성을 간소화하고 불명확한 문제의 속성을 재구성하고 문제의 역동성을 미리 예측하여 처리할 수 있는 능력 향상을 위한 교육 내용 및 방법으로 전환되어야 한다[4].

과제 중심 교수 전략은 학습자들이 계열화된 전체 과제 학습을 진행할 때 학습이 강화된다는 것으로 다루는 과제의 유형이 실제적 상황과 실세계 문제의 형태를 유지한다는 점에서 문제 중심 학습과 유사하지만, 문제 중심 학습과 달리 보다 직접적인 교수적 처치를 가한다는 점에서 차이가 있다[2]. 즉, 기존의 전형적인 문제 중심 교수 전략들은 학습자들을 협력적 소집단으로 구성하고 각 집단의 학습자들에게 해결해야 할 복잡한 문제와 문제 해결에 사용될 수 있는 자원들을 제공한다. 학습자들은 집단 구성원들과의 상호작용과 자원 검색 등의 과정을 통해 스스로 자신의 해법을 구성해나가는데 복잡한 문제는 학습자들 스스로 해결하기에 너무 난해하거나 어려울 수 있다. 따라서 과도한 시간과 노력의 투자가 요구

되며, 학습자 스스로 해법을 도출하는 과정에서 다양한 오개념들을 유발할 수 있다. 이에 따라 과거의 연구들은 이러한 개방형 문제 해결 전략이 비효율적일 뿐 아니라, 원하는 능력을 학습시키는데 비효과적이었음을 제시해 왔다[9].

3. 과제 중심 스크래치 프로그래밍 학습 프로그램 개발

3.1 설계 원리

과제 중심 스크래치 프로그래밍 학습 프로그램은 Merrill의 효과적인 교수의 5가지 핵심 원리를 토대로 설계하였으며 구체적인 설계 원리는 다음과 같다.

첫째, 전체 과제 중심 전략을 설계한다. 이는 Merrill이 제시한 효과적인 교수의 핵심 원리 중 과제 중심 원리를 토대로 한 것으로 학습은 학습자가 실제적 과제 중심 교수 전략에 참여할 때 증진된다는 것이다[2]. 학습자의 실제적이고 통합적인 문제해결능력 향상을 위해서는 실세계의 복잡한 문제해결경험을 학습자들에게 제공할 필요가 있다. 그러나 어린 학습자들은 자신이 학습한 모든 내용들을 통합하고 통합된 지식을 토대로 실생활의 과제들을 해결하는데 어려움을 겪는다[10][11]. 따라서 학습의 전이를 촉진하기 위해서는 통합된 목표에 도달하는데 중점을 두어야 하며 학습자들에게 전체적인 복잡한 기술을 연습할 기회를 제공해야 한다[12]. 그러나 단지 복잡한 전체 과제와 정보를 학습자들에게 아무런 교수적 개입 없이 제시하는 것은 비효율적이며, 학습자들에게 높은 인지적 부담을 유발할 수 있기 때문에 체계적인 과제 설계 및 제시 전략이 필요하다.

둘째, 과제와 관련된 인지 구조 활성화 전략을 설계한다. 이는 활성화의 원리를 토대로 한 것으로 학습은 학습자가 관련된 사전 지식이나 경험을 활성화할 때 증진된다는 것이다[2]. 즉 효과적인 교수의 첫 단계는 과제가 제시되면 학습자들이 해당 과제 수행을 위해 활용 가능한 사전 경험과 지식을 상기할 수 있도록 유도하는 것이다. 이 때 학습자의 오개념을 확인하고 정정하기 위

하 교수적 처치가 필요하며, 제한적인 사전 경험을 가지고 있다면 관련된 경험을 충분히 제공한다.

셋째, 사례 중심 학습 내용 시연 전략을 설계한다. 이는 시연의 원리를 토대로 한 것으로 학습은 학습자가 학습할 내용의 시연을 관찰할 수 있을 때 증진된다는 것이다[2]. 따라서 학습자의 과제 해결을 위해 필요한 지식을 제공하기 위해서는 일반적이고 포괄적인 형태의 정보를 제공하는 것이 아니라 보다 구체적인 사례를 통해 제공한다.

넷째, 반복적 학습 내용 적용 전략을 설계한다. 이는 적용의 원리를 토대로 한 것으로 학습은 학습자가 습득한 지식을 새로운 상황에 적용함으로써 증진된다는 것이다[2]. 따라서 학습자가 습득한 지식과 기술을 다양한 문제 상황에서 반복적으로 적용해 볼 수 있는 기회를 제공해야 하며 다양한 적용 과제와 함께 적합한 코칭 및 피드백을 제공한다.

다섯째, 효과적인 학습 내용 통합 전략을 설계한다. 이는 통합의 원리를 토대로 한 것으로 학습은 학습자가 새로운 지식을 자신의 일상 세계로 통합할 때 증진된다는 것이다[9]. 이러한 통합의 기회는 학습한 내용에 관한 유의미한 성찰의 기회와 실생활에서의 활용 기회, 공개적 시연의 기회를 제공함으로써 증진시킬 수 있다.

3.2 학습 단계 및 프로그램 개발

이러한 설계 원리를 토대로 개발한 학습 단계는 과제 확인, 지원, 과제 해결, 통합의 4단계로 이루어지며, 각 단계에서 교수자의 교수 활동은 전체 과제 및 수행 목표 제공, 과제 해결을 위해 필요한 지원 정보 제공, 절차 정보의 적시 제공 및 연습을 위한 부분 과제의 제공, 결과 성찰 및 공개적 시연의 기회 제공의 순서로 이루어진다. 이와 동시에 이루어지는 학습자들의 학습 활동은 과제 수행 목표 확인, 인지 구조 활성화, 시연 관찰 및 지식 적용을 통한 과제 해결, 지식 통합의 단계로 이루어진다. 학습 프로그램의 핵심 요소인 학습 과제, 지원 정보, 절차 정보, 부분 과제 연습의 유형과 제시 방법은 <표 1>과 같다.

<표 1> 핵심 요소 유형 및 제시 방법

구성 요소	유형 및 제시 방법	
학습 과제	유형	귀납적 추론 과정 촉진을 위한 전체 과제 실제적 과제 모델링 예제, 과정 활동지, 전통적 과제
	제시 방식	모델링 예제, 과정 활동지, 전통적 과제 순으로 제시
부분 과제 연습	유형	전체과제 수행을 위해 요구되는 순환적 기능 숙달을 위한 부분 과제 연습 과제
	제시 방식	전체과제를 먼저 제시한 후, 부분 과제 연습 문제 제시
지원 정보	유형	전문가 모델링 예제 교사의 설명에 의한 인지적 피드백
	제시 방식	다양한 멀티미디어를 활용한 정보의 제시 전문가 모델링의 경우 학습 과제 수행 직전에 제시 인지적 피드백은 과제 집합 전체를 수행 완료한 후 제시
절차 정보	유형	규칙과 절차 제공을 위한 시연 선수 지식 제공을 위한 실례
	제시 방식	전체과제 수행 중 필요시 적시 제공 과제와 정보를 한 화면에 통합적으로 제공 이전 학습 과제에서 제시된 절차 정보의 소거

이러한 설계 전략들에 따라 전체 6주(12차시) 분량의 학습 프로그램을 개발하였으며, 구성된 세부 학습 프로그램은 <표 2>와 같다.

4. 연구 방법

4.1 연구가설

연구 목적 달성을 위해 설정한 연구가설은 다음과 같다.

과제 중심 스크래치 프로그래밍 학습은 중학교 학습자의 문제해결력에 유의한 영향을 미친다.

4.2 연구대상

연구대상은 경기도 지역 중학교 1학년 학생들로 실험설계에 따라 실험집단1 33명, 실험집단2 31명, 비교집단 31명으로 구성하였다.

<표 2> 과제 중심 스크래치 프로그래밍 학습 프로그램

기간	학습 개념	학습 과제	지원 정보	절차 정보	부분 과제 연습
1주	순차 처리	스프라이트 동작 제어 프로그램 - 과제 1(모델링 예제): 이동하는 고양이 - 과제 2(과정 활동지): 회전하는 고양이 - 과제 3(전통적 과제): 달리는 고양이	순차 구조	동작 팔레트의 각 블록 사용법	전진 후진 우회전 좌회전
2주	순차 처리 조건 처리	스프라이트 형태 제어 프로그램 - 과제 1(모델링 예제): 말하고 생각하는 고양이 - 과제 2(과정 활동지): 모양 바꾸는 고양이 - 과제 3(전통적 과제): 그래픽 효과 적용	조건 구조	형태 팔레트의 각 블록 사용법 제어 팔레트의 조건 제어를 위한 각 블록 사용법	문자 출력 모양 변경 그래픽 효과
3주	반복 처리 다중 조건 처리	애니메이션 프로그램 - 과제 1(모델링 예제): 수족관 애니메이션 - 과제 2(전통적 과제): 슬라이드 쇼 애니메이션	반복 구조 다중 조건 구조	제어 팔레트의 반복 제어를 위한 각 블록 사용법 연산 팔레트의 논리 및 관계 연산 블록 사용법	조건 생성
4주	다중 반복 처리 변수	상호작용 게임 프로그램 - 과제 1(과정 활동지): 물고기 잡기 게임 - 과제 3(전통적 과제): 재사용품 분리수거 게임	다중 반복 구조 생성 방식 변수의 개념	변수 팔레트의 변수 블록 사용법	변수 생성 변수 삭제
5주	병렬 처리	시뮬레이션 프로그램 - 과제 1(과정 활동지): 소리 그래프 - 과제 2(완성형 과제): 소리 시뮬레이션	병렬 처리 구조	관찰 및 펜, 소리 팔레트의 각 블록 사용법	
6주	자료의 입출력 리스트	자료 처리 프로그램 - 과제 1(과정 활동지): 친구 명단 삽입 - 과제 2(과정 활동지): 친구 명단 정렬 - 과제 3(전통적 과제): 친구 명단 탐색	정렬 알고리즘 탐색 알고리즘	변수 팔레트의 리스트 블록 사용법 연산 팔레트의 각 블록 사용법	리스트 생성 리스트 삭제 리스트에 자료 삽입

4.3 실험설계 및 절차

연구의 실험설계 및 절차는 다음과 같다.

실험집단1	O ₁	X ₁	O ₂
실험집단2	O ₃	X ₂	O ₄
비교집단	O ₅	X ₃	O ₆

- O₁, O₃, O₅: 사전 문제해결력 검사
 X₁: 과제 중심 스크래치 프로그래밍 학습
 X₂: 전통적인 스크래치 프로그래밍 학습
 X₃: 전통적인 비주얼베이식 프로그래밍 학습
 O₂, O₄, O₆: 사후 문제해결력 검사

가장 먼저 사전 문제해결력 검사를 통해 3개 집단의 동질성 여부를 분석하였다. 이후, 실험집단1에는 과제중심 스크래치 프로그래밍 학습을 진행하고, 실험집단2에는 전통적인 스크래치 프로그래밍 학습을, 비교집단에는 비주얼 베이식을 활용한 전통적인 프로그래밍 학습을 실시하였다. 전통적인 프로그래밍 학습은 본 연구에서 설계한 과제 중심 교수 전략이 제거되고 단지 학습자에게 학습 내용을 전달하고 실습할 수 있는 환경을 제공하는 것을 의미한다. 단 세 집단 모두 동일한

프로그래밍 개념 습득을 위한 학습 내용들로 구성하였으며 12차시(6주)에 걸쳐 학습이 진행되었다. 실험 처치 직후, 사후 문제해결력 검사를 실시하여 실험 효과 여부를 분석하였다.

4.4 연구도구

문제해결력 검사를 위한 측정 도구는 2003년 OECD 학업성취도국제비교연구(PISA 2003)의 문제해결 영역 문항을 본 연구의 연구대상에 맞게 선정 및 수정하여 구성하였다[13]. PISA 2003의 문제해결 영역은 총 19개의 문항으로 구성되어 있으며 의사결정, 시스템 분석과 설계, 문제점 해결의 3가지 유형으로 구분된다. 각 문제에서 제시되고 있는 문제 상황은 가능한 한 실생활에서 접할 수 있는 상황으로 선정되었으며, 범교과적 차원에서의 문제해결력을 평가하기 위한 문항들로 구성되어 있기 때문에 영역 일반적인 문제해결력 측정에 유용하게 활용될 수 있다.

본 연구에서는 19개의 문항 중 각 유형별로 3 문항씩 선정하여 총 9개의 문항을 측정 도구로

사용하였으며, 본 연구의 연구대상인 중학교 1학년 학습자의 발달 수준 및 국내의 사회문화적 상황에 맞게 문제의 상황을 수정하였다. 구체적인 문항의 특성은 <표 3>과 같다.

검사 결과는 정답의 경우 2점, 오답의 경우 0점을 부여하였으며, OECD에서 공개된 문항채점기준에 따라 부분점수로 인정될 경우 1점을 부여하였다. 따라서 가능한 점수 분포는 최고 18에서 최저 0이다.

<표 3> 문제해결력 측정 문항

번호	주제	문제 유형	형태
1	에너지 요구량	의사결정	개방 서술형
2	도서관 관리체계	시스템 분석과 설계	개방 서술형
3	냉동고	문제점 해결	선택형
4	수에 의한 디자인	시스템 분석과 설계	선택형
5	수에 의한 디자인	시스템 분석과 설계	개방 서술형
6	영화 보러 가기	의사결정	선택형
7	환승 체계	의사결정	개방 서술형
8	관계	문제점 해결	개방 서술형
9	관계	문제점 해결	선택형

5. 연구결과 및 논의

실험처치 전 실시한 사전 문제해결력 검사 결과, <표 4>, <표 5>에서와 같이 실험집단1, 실험집단2, 비교집단은 사전 문제해결력 검사 총점에 있어서 통계적으로 유의한 차이가 없었으며($F=1.300$, $p=.277$), 각 문제 유형별로 세분화된 의사결정($F=.365$, $p=.695$), 시스템 분석과 설계($F=.200$, $p=.819$), 문제점 해결($F=3.080$, $p=.051$)의 세 영역에서도 세 집단 간에 유의한 차이가 나타나지 않았다.

<표 4> 사전 문제해결력 검사 결과

집단	의사결정	시스템 분석과 설계	문제점 해결	총점	
실험1 (N=33)	M	3.30	3.91	2.30	9.51
	SD	1.358	1.739	1.610	2.774
실험2 (N=31)	M	3.23	4.03	1.29	8.55
	SD	1.257	1.169	1.442	1.947
비교 (N=31)	M	3.03	4.16	1.97	9.16
	SD	1.278	1.791	1.888	2.437
전체 (N=95)	M	3.19	4.03	1.86	9.08
	SD	1.291	1.581	1.692	2.426

M: 평균, SD: 표준편차

<표 5> 세 집단 간의 사전 문제해결력 검사에 대한 차이검증(ANOVA)

		SS	df	MS	F	p
의사결정	집단 간	1.233	2	.616	.365	.695
	집단 내	155.357	92	1.689		
	합계	156.589	94			
시스템 분석과 설계	집단 간	1.017	2	.508	.200	.819
	집단 내	233.889	92	2.542		
	합계	234.905	94			
문제점 해결	집단 간	16.897	2	8.448	3.080	.051
	집단 내	252.325	92	2.743		
	합계	269.221	94			
총점	집단 간	15.213	2	7.606	1.300	.277
	집단 내	538.113	92	5.849		
	합계	553.326	94			

p < .05, SS:제곱합, MS:평균제곱

실험처치 후, 사후 문제해결력 검사를 실시하였으며, 세 집단 간의 사후 문제해결력 차이검증 결과는 <표 6>, <표 7>, <표 8>과 같다.

<표 6> 사후 문제해결력 검사 결과

집단	의사결정	시스템 분석과 설계	문제점 해결	총점	
실험1 (N=33)	M	4.03	4.85	4.33	13.21
	SD	1.045	1.564	1.339	2.655
실험2 (N=31)	M	3.74	4.71	3.16	11.61
	SD	.815	1.553	1.951	2.654
비교 (N=31)	M	3.35	4.94	2.23	10.52
	SD	1.170	1.263	1.687	2.502
전체 (N=95)	M	3.72	4.83	3.26	11.81
	SD	1.048	1.456	1.869	2.811

M: 평균, SD: 표준편차

<표 7> 세 집단 간의 사후 문제해결력 검사에 대한 차이검증(ANOVA)

		SS	df	MS	F	p
의사결정	집단 간	7.324	2	3.662	3.510*	.034
	집단 내	96.002	92	1.043		
	합계	103.326	94			
시스템 분석과 설계	집단 간	.805	2	.402	.186	.830
	집단 내	198.500	92	2.158		
	합계	199.305	94			
문제점 해결	집단 간	71.475	2	35.737	12.796*	.000
	집단 내	256.946	92	2.793		
	합계	328.421	94			
총점	집단 간	117.978	2	58.989	8.689*	.000
	집단 내	624.612	92	6.789		
	합계	742.589	94			

*p < .05, SS:제곱합, MS:평균제곱

<표 8> 집단 간 사후 문제해결력 검사에 대한 사후 차이검증(Post Hoc Tests: Tukey)

	(i)해당 집단	(j)상대 집단	평균차(i-j)	표준 오차	p
의사결정	실험1	실험2	.29	.256	.499
		비교	.68*	.256	.026
시스템 분석과 설계	실험1	비교	.39	.259	.300
	실험2	비교	.14	.367	.924
문제점 해결	실험1	비교	-.09	.367	.970
	실험2	비교	-.23	.373	.818
총점	실험1	비교	1.17*	.418	.017
	실험2	비교	2.11*	.418	.000
	실험1	비교	.935	.424	.076
	실험2	비교	1.60*	.652	.042
	실험1	비교	2.70*	.652	.000
	실험2	비교	1.10	.662	.227

*p < .05

<표 6>에서와 같이 사후 문제해결력 검사 총점은 실험집단1(M=13.21), 실험집단2(M=11.61), 비교집단(M=10.52) 순으로 높게 나타났으며, <표 7>에서와 같이 세 집단 간 점수 차이가 통계적으로 유의한 것으로 드러났다(F=8.689, p=.000). 각 문제 유형별로 세분화된 영역별로 살펴보면, 의사결정(F=3.510, p=.034)과 문제점 해결(F=12.796, p=.000) 영역의 경우 유의한 차이가 나타났으나, 시스템 분석과 설계(F=.186, p=.830) 영역의 경우 유의가 차이가 나타나지 않았다.

또한 <표 8>에 제시된 사후검증 분석 결과에 따르면 실험집단1이 실험집단2(평균차=1.60, p=.042)와 비교집단(평균차=2.70, p=.000)보다 유익하게 문제해결력 총점이 높은 것으로 드러났으며,

실험집단2와 비교집단은 유의한 차이가 나타나지 않았다(평균차=1.10, p=.227). 각 문제 유형별로 세분화된 영역별로 살펴보면, 의사결정 영역은 실험집단1의 평균이 비교집단(평균차=.68, p=.026)보다 유의하게 높게 나타났으며, 문제점 해결 영역은 실험집단1의 평균이 실험집단2(평균차=1.17, p=.017)와 비교집단(평균차=2.70, p=.000)보다 유익하게 높게 나타났다. 반면, 시스템 분석과 설계 영역의 경우, 세 집단 간의 유의한 차이가 나타나지 않았다. 마지막으로 각 집단별 문제해결력 향상 정도가 통계적으로 유의한지 알아보기 위해 대응표본 t검정을 실시하였으며 그 결과는 <표 9>, <표 10>, <표 11>과 같다.

<표 9> 실험집단1의 문제해결력 향상 정도

	사전 문제해결력 - 사후 문제해결력				
	M	SD	t	df	p
의사결정	-0.73	1.506	-2.775*	32	.009
시스템분석과 설계	-0.94	1.952	-2.765*	32	.009
문제점 해결	-2.03	1.741	-6.700*	32	.000
총점	-3.70	2.778	-7.645*	32	.000

*p < .05, M:평균, SD: 표준편차

<표 10> 실험집단2의 문제해결력 향상 정도

	사전 문제해결력 - 사후 문제해결력				
	M	SD	t	df	p
의사결정	-.52	1.313	-2.188*	30	.037
시스템분석과 설계	-.68	1.922	-1.963	30	.059
문제점 해결	-1.87	2.349	-4.435*	30	.000
총점	-3.06	2.394	-7.129*	30	.000

*p < .05, M:평균, SD: 표준편차

<표 11> 비교집단의 문제해결력 향상 정도

	사전 문제해결력 - 사후 문제해결력				
	M	SD	t	df	p
의사결정	-.32	1.641	-1.095	30	.282
시스템분석과 설계	-.77	1.857	-2.322*	30	.027
문제점 해결	-.26	1.413	-1.017	30	.317
총점	-1.35	1.473	-5.121*	30	.000

*p < .05, M:평균, SD: 표준편차

과제중심 스크래치 프로그래밍 학습 집단인 실험집단1의 경우, 문제해결력 총점 뿐 아니라 모든

영역에서 점수가 향상되었으며 이러한 점수의 향상이 통계적으로 유의한 것으로 나타났다. 전통적인 스크래치 프로그래밍 학습 집단인 실험집단2 또한 마찬가지로 문제해결력 총점이 유의하게 향상되었으나, 영역별로 살펴보면 의사결정과 문제점 해결 영역이 유의하게 향상되었다. 마지막으로 전통적인 비주얼 베이식 프로그래밍 학습 집단인 비교집단의 경우 문제해결력 총점이 유의하게 향상되었으나 시스템 분석과 설계 영역에서만 유의한 향상을 보이고 있다.

6. 결 론

문제해결력 향상을 위한 프로그래밍 교육의 목표는 프로그래밍 도구 자체의 사용법을 습득하는 차원을 넘어 프로그래밍 과정을 통한 실제적 문제해결경험을 제공하는데 있다. 문제해결과정을 설계하고 이를 컴퓨터 프로그램으로 구현하는 프로그래밍의 과정은 하나의 복잡한 문제해결과정으로 이러한 과정을 경험함으로써 다양한 지식과 기술을 통합하고 문제 상황에 유연하게 대처하기 위한 전략적 지식을 습득할 수 있다. 따라서 이러한 문제해결력 향상을 지원하기 위해 프로그래밍 교육이 필요하지만 프로그래밍 도구 활용 방법의 학습으로 인해 발생할 수 있는 학습 저해 요소들을 최소화할 필요가 있다.

스크래치와 같은 교육용 프로그래밍 언어를 사용할 경우 도구 사용으로 인한 학습자의 인지적 부담을 감소시키고 실생활의 문제해결과정과 유사한 경험을 제공할 수 있으며, 학습자가 프로그래밍 과정을 통한 문제해결에 집중할 수 있도록 할 수 있다. 그러나 새로운 도구의 활용이 완전한 학습효과를 보장할 수 없다.

따라서 본 연구에서는 문제해결력 향상을 위한 과제 중심 스크래치 프로그래밍 학습 프로그램을 개발하였다. 해당 프로그램은 중학교 1학년 학습자를 대상으로 하고 있으며 전체 과제 중심 교수 전략을 지향한다. 여기서 과제는 실제적 문제해결 학습을 위해 설계된 학습 과제로 복잡한 문제의 속성을 지닌다. 학습 프로그램의 단계는 Merrill이 제시한 교수의 5가지 핵심 원리를 토대로 구성하였으며, 프로그램의 핵심 요소인 과제와 정보의

유형 및 제시방식은 4CID 모델을 토대로 설계하였다. 개발된 학습 프로그램의 효과성과 더불어 스크래치 활용의 효과성을 확인하기 위해 중학교 1학년 학습자들을 대상으로 프로그램을 적용하고 문제해결력에 미치는 영향을 분석하였다.

연구 결과를 종합적으로 논의해보면 다음과 같다.

첫째, 프로그래밍 학습의 효과성을 확인하였다. 연구 결과 세 집단 모두 유의한 문제해결력의 향상을 나타내고 있다. 이는 프로그래밍 도구나 교수법에 상관없이 기본적으로 프로그래밍 학습이 학습자의 문제해결력 향상에 긍정적인 영향을 주는 것으로 해석할 수 있다.

둘째, 과제 중심 스크래치 프로그래밍 학습 설계의 효과성을 확인하였다. 실험집단1과 실험집단2 모두 스크래치를 활용한 프로그래밍 학습을 진행하였지만, 과제 중심 교수 학습 설계를 적용한 실험집단1의 경우, 문제해결력의 모든 영역에서 유의한 향상을 보이고 있다. 또한 사후 문제해결력 검사의 집단별 차이검증 결과에서도 알 수 있듯이, 실험집단1의 사후 문제해결력은 세 집단 중 유의하게 가장 높으며, 특히 문제점 해결 영역의 경우, 다른 두 집단보다 유의하게 높게 나타났다. 이는 과제 중심 교수 학습 설계가 학습자의 문제해결력 향상에 긍정적인 영향을 주었으며, 특히 문제점 해결 능력 향상에 긍정적인 영향을 준 것으로 해석할 수 있다. 이러한 결과는 또한 도구의 전환이 학습 효과를 보장하는 것은 아니라는 점을 보여준다.

이러한 연구 결과를 바탕으로, 향후 연구에서 본 연구를 통해 제시된 교수 전략들 중 어떤 요인이 학습자의 문제해결력에 유의한 영향을 끼치는지에 관한 세부적인 분석 및 관련성 탐구에 관한 연구가 필요할 것이다.

참 고 문 헌

- [1] Jonassen, D. H., Howland, J., Moore, J., & Marra, R. M.(2003). *Learning to Solve Problems with Technology: A Constructivist Perspective*. NJ: Merrill Prentice Hall.
- [2] Merrill, M. D.(2007). *First principles of*

- instruction: A synthesis. In Reiser, R. A., & Dempsey, J. V. (Eds.), *Trends and Issues in Instructional Design and Technology 2nd Edition*(62-71). NJ: Prentice Hall.
- [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. 박사학위논문, 한국교원대학교.
- [5] 이은경, 이영준(2008). Scratch 활용 프로그래밍 교육이 중학생의 몰입수준과 프로그래밍 능력에 미치는 영향. *중등교육연구*, 56(2), 359-382.
- [6] 정미연, 이은경, 이영준(2008). Squeak Etoys 활용 알고리즘 학습이 중학생의 문제해결력에 미치는 영향. *대한공업교육학회지*, 33(2), 170-191.
- [7] Dörner, D.(1996). *The Logic of Failure: Recognizing and Avoiding Error in Complex Situations*. MA: Addison-Wesley.
- [8] Wenke, D., & Frensch, P. A.(2003). Is Success or Failure at Solving Complex Problems Related to Intellectual Ability?. In Davidson, J. E., & Sternberg, R. J.(Eds.), *The Psychology of Problem Solving*(pp.87-126). NY: Cambridge University Press.
- [9] Kirschner, P. A., Sweller, J., & Clark, R. E.(2006). Why minimal guidance during instruction does not work: an analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2), 75-86.
- [10] Ragan, T. J., & Smith, P. L.(1996). Conditions theory and models for designing instruction. In D. Jonassen (Ed.), *Handbook of research on educational communications and technology*(2nd ed.)(623-650). Mahwah, NJ: Lawrence Erlbaum Associates.
- [11] Spector, J. M., & Anderson, T. M. (Eds.) (2000). *Holistic and integrated perspectives on learning, technology, and instruction: Understanding complexity*. Mahwah, NJ: Lawrence Erlbaum Associates.
- [12] Achtenhagen, F.(2001). Criteria for the development of complex teaching-learning environment. *Instructional Science*, 29, 361-380.
- [13] 이미경, 광영순, 민경석, 채선희, 최성연, 나귀수, 박경미(2004). OECD/PISA 2003 평가틀 및 예시문항 - 수학, 과학, 문제해결력 소양 영역 -. 연구자료 ORM 2004-25-1, 한국교육과정평가원.



이 은 경

1998 한국교원대학교
컴퓨터교육과(교육학학사)

2009 한국교원대학교
컴퓨터교육과(교육학박사)

현재 경기 청운중학교 교사

관심분야: 정보교육, 로봇프로그래밍, 학습과학

E-Mail: soph76@hitel.net