JDCS 디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 20, No. 5, pp. 955-961, May 2019

비SW전공자 대상 PBL기반 프로그래밍 교육의 효과성 연구

고 광 일
우송대학교 테크노미디어융합학부 영상콘텐츠전공

A Study on the Effectiveness of PBL-based Programming Education for non-SW Major

Kwangil KO
Video Contents Major, School of Techno-Media Convergence, Woosong University, Daejeon, 34606, Korea

[요 약]

모든 산업분야에 SW가 긴밀히 융합되는 4차 산업혁명시대를 대응하기 위해 비SW전공 대상으로 컴퓨팅 사고력, 프로그래밍 등의 SW교육이 확대되고 있다. 본 연구는 비SW전공 대상 프로그래밍 교육에 문제중심학습 (PBL) 방식을 적용할 때 컴퓨팅 사고력과 학업 성취도에 미치는 효과를 분석하였다. 이를 위해 비SW전공 대상 프로그래밍 수업을 PBL 기반의 프로그래밍 교육과정 적용 반과 비적용 반으로 나누고, 수업 종료 후 두 반의 시험 성적과 학습 효과성 및 컴퓨팅 사고력에 대한 설문조사 결과를 분석하였다. 실험 결과, PBL 적용 반 학생들이 비적용 반 학생들에 비해 알고리즘 구현 능력, 자기주도적 학습, 문제해결능력, 지속적인 학습동기 항목에서 두드러지게 높은 점수를 받았고, 컴퓨팅 사고력에 대한 대부분의 설문 항목에서 통계적으로 유의미하게 높은 점수를 받았다. 이런 교육효과는 자발적 SW학습이 지속적으로 요구되는 4차 산업혁명시대에서 의미하는 바가 크다.

[Abstract]

In order to cope with the fourth industrial revolution era, in which SW plays an important role in all industries, the SW educations such as computational thinking and programming skill are being expanded to non-SW majors. The purpose of this study is to analyze the effects of the problem-based learning (PBL) on the computational thinking and academic achievement. To accomplish this, we divided a programming class of non-SW major into the PBL-based programming curriculum applied class and non-applied class. At the end of the classes, we analyzed the results of the exams and the questionnaire survey on the learning effectiveness and the computational thinking. As a result, the students in the PBL-applied class showed significantly higher scores in algorithm implementation ability, self-directed learning, problem solving ability, and continuous learning motivation. The educational effects are significant in the fourth industrial revolution era, where voluntary SW learning is continuously required.

색인어 : 프로그래밍, 문제중심학습, 컴퓨팅 사고력, 학업 성취도, 비SW전공
Key word : Programming, Problem-Based Learning, Computational Thinking, Academic Achievement, non-SW Major

http://dx.doi.org/10.9728/dcs.2019.20.5.955

Received 20 April 2019; **Revised** 15 May 2019
Accepted 27 May 2019

***Corresponding Author; Kwangil KO**
Tel: +82-42-630-9343
E-mail: kwangil.ko@gmail.com

Ⅰ. Introduction

Programming, which is the core of Software (SW) education, is used as a useful tool to teach not only the skill for understanding and developing software, but also the major components of the computational thinking such as abstraction, decomposition, pattern recognition, and algorithm. However, programming is recognized as a difficult course even for SW-major students due to the difficulty of grammar and the demands on the mathematical and logical thinking skills. In order to minimize the difficulties of the programming education, the studies have been conducted mainly focusing on utilizing educational programming languages (EPL) for the secondary education students[1-3]. In universities, the importance of SW convergence education is increasing and the researches for improving the programming ability and computational thinking of the SW and/or non-SW major students[4-6].

Problem Based Learning (PBL) is a learning method and/or an environment in which the flexible knowledge utilization, effective problem solving ability, self-directed learning ability, and effective collaboration ability are cultivated in the process of solving the unstructured and practical problems[7]. By the PBL, the student leads the collaborative activities from collecting the data to designing the solution for the given problem, and the instructor's role is an assistant to observe and guide the learning process. PBL originated in medical education and currently used in various fields to help students develop the practical problem solving skills and intrinsic motivation for learning. In the situation where the dependence of private tutoring and passive learning attitude are deepening, the internalization of self-directed learning ability is recognized as a major task of university education and the PBL is attracting interest by the advantages described above.

In this study, we applied PBL to the programming education for non-SW major students and analyzed the effects on the academic achievement and computational thinking. To do this, we designed a PBL-based programming curriculum for non-SW major students and devised the questionnaire to survey the effectiveness on their learning and computational thinking. Considering that the participants of the programming course were non-SW major, we used Scratch[8], which is one of the representative EPL's. The problems used in PBL were classified into two types. One is algorithm implementation problem and the other is creative SW production problem. The problems are presented to the students according to the class progress. To analyze the effectiveness of the PBL-based programming education, we divided a programming course into two classes: One class adopted the PBL-based programming curriculum and the other class didn't. At the end of the course, the scores of the exams and the results of the survey on the educational effects and computational thinking of the two classes were analyzed.

Ⅱ. Related Works

2-1 Computational Thinking

The computational thinking, emphasized by Wing (2006), refers to the basic thinking skills based on the computer science principles that all people in the 21st century should have. There are various definitions for the components of the computational thinking. Wing (2006) distinguishes the computational thinking as the abstraction and automation abilities[9] and the Computer Science Teachers Association (CSTA) has subdivided it into the data collection/analysis/ representation, problem decomposition, abstraction, algorithm, automation, simulation, and parallelization[10]. The Korean Ministry of Education replaced, in the definition of CSTA, data representation with structuring, abstraction with modeling, and included coding and simulation in automation. In order to solve the problems of teaching and learning activities caused by too granular components, KERIS proposed a DPAA(P) model consisting of five elements: decomposition(D), pattern recognition(P), abstraction(A), algorithm(A), and programming (P)[11].

In the age when all industries are converged with software and computers, the computational thinking is the key goal of the programming education for non-SW major students.

2-2 Problem-Based Learning (PBL)

PBL, which started in medical education, is spreading to various fields as a learner-centered learning model in which learners solve the presented practical problems. Through PBL, students can acquire a variety of information to solve problems, internalize learning motivation, and gain effective problem-solving abilities, self-directed learning, and effective collaborative abilities.

Several researches on the effects of PBL have been conducted. The effects of PBL on the logical thinking and problem solving abilities of the under high school students have been studied[12,13] and the effects of applying PBL to various college subjects have been studied[14,15]. Recently, the studies on applying PBL to programming education have also been carried out[16,17].

Since the SW education is gradually spreading to non-SW majors, it is necessary to study the effects of PBL applied programming education on the academic achievement and the computational thinking abilities for non-SW major students.

Ⅲ. PBL-Based Programming Education

3-1 PBL-Based Programming Curriculum

The programming class in this study was a liberal arts class that was held during one semester and conducted for 2 hours in a week for a total of 15 weeks. Because the students in the class were unfamiliar with programming, we used the 'Partial-PBL' method, which conducted Scratch training in the first three weeks and then applied PBL to the class, rather than the 'Whole-PBL' method in which the entire semester was conducted as PBL.

Two types of problems were presented to students. Students resolved the problem of the type-1 from the 4th week to the 7th week in the 'Mixed-PBL' approach where the professor lectured basic programming techniques for one hour and the students performed PBL for the remaining one hour. The problems of the type-2 were resolved from 9th week to 14th week in the 'Pure-PBL' approach where the intervention of the professor was minimized and the students conducted PBL for 2 hours. <Table 1> shows the curriculum of the PBL-based programming class for 15 weeks.

3-2 Problems used in PBL

The problems used in the PBL were classified into two types: the problem of the type-1 mainly requires the abilities for designing and implementing the algorithms to solve the problem, and the problem of the type-2 requires the activities for planing creatively SW contents and programming them in Scratch. More specifically said, the problem of the type-1 is to implement a program for analyzing the test result of a class. The program should have the functions of calculating the lowest, highest, median, and average values. To solve the problem, students were guided to investigate and understand the typical sorting algorithms of O(n2) - bubble, insert, and select sorts, and use one of them for problem solving. The problem of the type-2 is that students choose one of game or interactive story, and implement it in Scratch. As there were no tip about the contents of the game and story, students should start from the planning activities such as the market data and consumer’s trend data collection and analysis. <Table 2> summarizes the problems of each type used in the PBL.

표 1. PBL기반 프로그래밍 교육 과정
Table 1. PBL-based programming curriculum

N	Activity	Description
1	Lecture	• Orientation including the introduction to PBL
2	Lecture	• How to start Scratch and use blocks
3	Lecture	• Sprite and stage motion, looks, sound
4	Lecture	• Programming concept, data and variables
	PBL Activity	• Understanding the problem of Type 1
5	Lecture	• Logic operation, repeat and select controls
	PBL Activity	• Information collection for problem solving
6	Lecture	• Data structure, procedure, parallelism
	PBL Activity	• Program Implementation
7	Lecture	• Event handling, sensing condition
	PBL Activity	• Present the results
8	Mid Exam	
9	PBL Activity	• Understanding the problem of Type 2
10	PBL Activity	• Information collection and analysis
11	PBL Activity	• Planning the game or interactive story
12	PBL Activity	• Programing game or interactive story 1
13	PBL Activity	• Programing game or interactive story 2
14	PBL Activity	• Present the results
15	Final Exam	

표 2. PBL에 제시된 프로그래밍 문제
Table 2. Programming problems used in PBL

Type	Problem	Description
Type 1	Test Result Analysis	Write a program to find the highest, lowest, average, and median of the 30-person test score. Understand bubble sort, insert sort, select sort, and use one of the algorithms for problem solving.
Type 2	Game	Create a casual game that can be implemented with Scratch. Investigate game genres and game properties, and plan a game that can be implemented with Scratch events and sensing functions.
	Interactive Story	Create an interactive story consisting of 30 scenes in which contents are branched by interaction. Plan the contents of children's preferences and investigate the transition techniques between scenes in the movie.

Ⅳ. Experiment Method

4-1 Preliminary Work for Experiment

For the experiment, we divided a programming course for non-SW major students of a local private university into two classes. One class (called ‘PBL Class’) was composed of 36 students and applied the PBL-based programming curriculum described in the previous chapter III and the other class

표 **3.** 실험 전 사전 설문조사
Table 3. Survey before the experiment

Survey Contents	PBL Class (36 Students)	Non-PBL Class (34 Students)
Have you ever studied computational thinking?	- Yes: 0 - No: 36	- Yes: 0 - No: 34
Have you ever learned Scratch?	- Yes: 1 - No: 35	- Yes: 1 - No: 33
Have you ever studied programming languages other than Scratch?	- Yes: 0 - No: 36	- Yes: 0 - No: 34

표 **4.** PBL기반 프로그래밍 교육의 효과성 측정 방법
Table 4. Measuring method of the effectiveness of PBL-based programming education

Effectiveness Type		Measuring Method
Academic achievement		
	Test score evaluation	Carry out mid-term and final exams using the same questions in the PBL and non-PBL classes.
	Learning effectiveness	Perform a survey consisting of 10 questions on the self-directed learning, continuous learning motivation, and so on.
Computational thinking		
	Cognitive level	Perform a survey consisting of 3 questions on the comprehension, interest, and necessity of computational thinking.
	Problem-solving ability	Perform a survey consisting of 6 questions on understanding, analyzing, and solving problems using computational thinking.

(called ‘non-PBL Class’) was composed of 34 students and taught with traditional lectures using a Scratch textbook.

We got the students' consent in advance about the difference in the way of teaching depending on the classes and surveyed to see the experiences of the students in the programming languages and computational thinking. <Table 3> summarizes the results. No student had experienced the computational thinking and programming languages other than Scratch, and there was one student trained with Scratch in each class.

4-2 Experiment Method and Data Processing

This study measured the effects of PBL-based programming education on the academic achievement and computational thinking (see <Table 4>). The academic achievement was again subdivided into the test score evaluation and the learning effectiveness. The test score was measured by carrying out the mid-term and final exams using the same questions in PBL and non-PBL classes. In order to measure the learning effectiveness, a survey consisting of 10 questions was conducted on self-directed learning ability, continuous learning willpower, and mutual cooperation. In order to measure the effects on computational thinking, a survey consisting of 9 questions was performed on the cognitive level of computational thinking and the problem solving ability using computational thinking. The responses of all questionnaires consisted of 5 points of Likerttechnique, and the results of the survey were analyzed using the IBM SPSS Statistics program.

표 **5.** PBL반 (36명), 비-PBL반 (34명)의 문제 유형별 성적에 대한 t-검정 분석
Table 5. t-test of the test scores by the problem types

Type	Class	M	SD	t	p
Concept and grammar	PBL	74.5	12.6	-1.74	0.0865
	non-PBL	79.4	11.3		
Simple code writing	PBL	66.8	15.7	-0.15	0.8791
	non-PBL	67.4	16.0		
Algorithm and programming	PBL	63.6	14.1	2.29	0.0247
	non-PBL	54.3	19.4		

Ⅴ. Experiment Result

5-1 Effectiveness on Academic Achievement

The problems of the mid-term and final exams were classified into the types of ‘concept and grammar’, ‘simple code writing’, and ‘algorithm and programming.’ <Table 5> shows the average scores of the exams of the PBL and non-PBL classes according to the type of problem. There was no significant difference between two classes regarding the 'concept and grammar' and 'simple code writing' types. However, for the 'algorithm and programming' type, the score of the PBL class was statistically significantly higher than that of the non-PBL class. It can be interpreted that the ability of programming algorithms is formed relatively high when performing self-directed learning with clear learning motivation.

<Table 6> shows the questionnaire items used to survey the learning effectiveness and the t-test results of the survey. The questionnaire items were designed based on NASEL, which is the criterion of teaching and learning competency of Korea Education Development Institute. According to the table, the PBL class showed statistically significant higher scores than the non-PBL class in all questionnaire items except for the first item asking the overall satisfaction level. In particular, self-directed learning, problem-solving ability, and motivation for learning have shown notable differences. This result is significant for students who have to live in the 4th industrial revolution era, which requires

표 6. PBL반 (36명), 비-PBL반 (34명)의 학습 효과성에 대한 t-검정 분석
Table 6. t-test of the learning effectiveness survey results

Item	Class	M	SD	t	p
Overall satisfied with the class?	PBL	4.69	0.63	1.48	0.150663
	non-PBL	4.23	0.92		
Actively participate in class?	PBL	4.91	0.10	5.35	4.32E-05
	non-PBL	3.63	0.69		
Awareness of programming improved?	PBL	4.69	0.63	3.91	0.000665
	non-PBL	3.61	0.77		
Intellectual curiosity stimulated?	PBL	4.77	0.44	4.46	0.000162
	non-PBL	3.69	0.75		
Collaboration with others successful?	PBL	4.69	0.48	3.82	0.000815
	non-PBL	3.77	0.72		
Thinking skills improved?	PBL	4.85	0.37	2.82	0.009420
	non-PBL	4.15	0.80		
Confident in programming skills?	PBL	4.61	0.50	4.37	0.000204
	non-PBL	3.62	0.65		
Self-directed learning achieved?	PBL	4.69	0.48	5.43	1.39E-05
	non-PBL	3.46	0.66		
Problem-solving skills improved?	PBL	4.52	0.51	5.05	3.54E-06
	non-PBL	3.85	0.61		
Motivation to learning established?	PBL	4.64	0.47	6.07	6.32E-08
	non-PBL	3.91	0.51		

voluntary and continuous learning of the software.

5-2 Effectiveness on Computational Thinking

In order to examine the effectiveness on the computational thinking, we conducted a questionnaire survey to investigate the cognitive level of computational thinking and the problem-solving ability using computational thinking.

<Table 7> shows the questionnaire items used to survey the effectiveness on the computational thinking and the t-test results of the survey. In the case of the cognitive level of computational thinking, the PBL class showed statistically significantly higher scores in all items than the non-PBL class. In particular, the score of the PBL class for the item of asking if the computational thinking is necessary was significantly higher than that of the non-PBL class. This can be interpreted as a result of experiencing the effects of computational thinking on actual problem solving activities in PBL.

In the case of the problem solving ability using the computational thinking, the scores of the PBL class was statistically significantly higher than those of non-PBL class in all items except one item. The difference between the two classes was small in the item of applying solutions to similar problems. It can be interpreted that the students in the PBL class did not experience similar problem solving before this study. It is noticeable that the score of the PBL class in the item of being confident in applying it to everyday life is significantly higher.

표 7. PBL반 (36명), 비-PBL반 (34명)의 컴퓨팅 사고력에 대한 t-검정 분석
Table 7. t-test of the computational thinking survey results

Item	Class	M	SD	t	p
Cognitive Level of Computational Thinking					
I feel that I understand CT	PBL	4.00	0.53	2.49	0.015212
	non-PBL	3.62	0.74		
I am Interested in CT	PBL	4.22	0.68	2.71	0.008623
	non-PBL	3.79	0.64		
I think CT is necessary	PBL	4.31	0.58	4.33	4.99E-05
	non-PBL	3.68	0.64		
Problem-Solving Ability using Computational Thinking					
Collecting the necessary data	PBL	4.25	0.50	6.63	6.44E-09
	non-PBL	3.41	0.56		
Simplifying problem with abstract concept	PBL	3.72	0.45	5.47	6.91E-07
	non-PBL	3.08	0.51		
Break the problem into small ones	PBL	3.83	0.61	2.60	0.011395
	non-PBL	3.41	0.74		
Apply a solution to similar problems	PBL	3.78	0.59	0.07	0.938914
	non-PBL	3.76	0.81		
Designing problem-solving algorithms	PBL	4.06	0.63	2.62	0.009409
	non-PBL	3.62	0.74		
Confident in applying CT to everyday life	PBL	4.14	0.54	4.63	1.70E-05
	non-PBL	3.47	0.66		

Ⅴ. Conclusion

This study analyzed the effects of PBL on the improvement of academic achievement and computational thinking in the non-SW major students' programming course. For this purpose, we have designed a PBL-based Scratch programming curriculum and carried out the PBL activities by presenting the feasible problems according to the progress of the course. To experimentally validate the effectiveness of PBL, we divided the course into the PBL class (of 36 students) and non-PBL class (of 34 students), and applied the PBL-based curriculum only to the PBL class. At the end of the classes, the scores of the exams and the results of the questionnaire survey on the learning effectiveness and computational thinking of the two classes were analyzed.

In conclusion, the questionnaire scores on the learning effectiveness and computational thinking of the PBL class were statistically significantly higher than those of the non-PBL class. The results of the experiment showed that the students in the PBL class had more self-directed learning and more motivation for continuous learning than those in the non-PBL class, and that they were more confident that they could apply the computational thinking to everyday life problems. In addition, we found that the students in the PBL class who experienced the practical problem-solving activities had a relatively superior ability to use algorithms.

It is important to note that the scores of the questionnaire items on the abstraction concept are significantly lower than those of the other items. This phenomenon, which is in common with the PBL class and non-PBL class, suggests that the research for improving abstraction ability is further needed in programming education.

This paper is different from other researches in that it has broadly analyzed the impact of PBL learning on two independent samples with respect to grades, learning effects, and computational thinking. Also, since the students of the two groups may have different qualities and tendencies, we plan to repeat the same experiment continuously to strengthen the objectivity.

Acknowledgment

This research is based on the support of 『2019 Woosong University Academic Research Funding』

References

[1] EunKyoung Lee, “A Task Centered Scratch Programming Learning Program for Enhancing Learners’ Problem Solving Abilities,” *The journal of Korean Association of Computer Education*, Vol. 12, No. 6, pp. 1-9, Nov. 2009.

[2] Hee Jin Noh and Seoung Hye Paik, “Students’ Perception of Scratch Program using High School Science Class,” *The Journal of Korean Association for Science Education*, Vol. 35, No. 1, pp. 53-64, 2015.

[3] Tae-Hun Kim, JongHoon Kim, “Development and implementation of STEAM Program based on Scratch Programming,” *The journal of Korean Association of Computer Education*, Vol. 17, No. 6, pp. 49-57, Nov. 2014.

[4] Su-Young Pi, “A Study on Coding Education of Non-Computer Majors for IT Convergence Education,” *The Journal of Digital Convergence*, Vol. 14, No. 10, pp. 1-8, Oct. 2016.

[5] Mi-Ja Oh, “Non-Major Students' Perceptions of Programming Education Using the Scratch Programming Language,” *The journal of Korean Association of Computer Education*, Vol. 20, No. 1, pp. 1-11, Jan. 2017.

[6] Kwangil KO, “A Study on the EPL using Instructional Model of SW Major’s Programming Class,” *The Journal of Digital Contents Society*, Vol. 19, No. 5, pp. 891-898, May. 2018.

[7] Cindy E. Hmelo-Silver, “Problem-Based Learning: What and How Do Students Learn?”, *Educational Psychology Review*, Vol. 16, Issue 3, Sep. 2004.

[8] Scratch web-site. Available: https://scratch.mit.edu/.

[9] Jeannette Wing, “Computational Thinking”, *Communications of the ACM,* Vol. 49, No. 3, 2006.

[10] CSTA web-site. Available: https://www.csteachers.org/.

[11] Jinsook Kim and et. al., “A Study on Instructional Model of SW,” Korea Educational Development Institute, CR 2015-35, 2015.

[12] Inae Kang, Jiyeon Yook, “A Case Study of A PBL-Based Design Class for Creative Personality Education”, *Journal of SAEK*, Vol. 30, No. 39, pp. 23-53, 2011.

[13] Soe Hye Kyoung, Kim You Me, “The effectiveness of a PBL based self-directed learning program”, *Journal of KALCI*, Vol. 12, No. 1, pp. 183-204, 2012.

[14] Kong Ha Rim, “A study on Applying Problem-based Learning Methods in College-level Composition Courses”, *Journal of Korean Culture*, Vol. 25, pp. 7-42, 2014.

[15] Kim Mi Young, Ryu Young Tae, “Problem-based learning in business manner class: A case study focusing on educational effectiveness”, *Journal of KASS*, Vol. 23, No. 2, pp. 5-25, 2014.

[16] SeongKyun Jeon, YoungJun Lee, “Design of Programming Education with App Inventor”, *Proceedings of KCI*, Vol. 22, No. 1, pp. 237-240, 2014.

[17] Young-Shin Han, “Effectiveness of problem-based learning based programming education : Focus on Computational Thinking”, *Journal of Multimedia Services Convergent with Art, Humanities, and Sociology*, Vol. 8, No. 7, 2018.

고광일(Kwangil KO)

1995년 : 포항공과대학교 전자계산학과 (학사, 석사)
1999년 : 포항공과대학교 컴퓨터공학과 (공학박사)
1999년~2010년 8월: (주)알티캐스트 사업품질관리본부 본부장 및 서비스개발사업팀 팀장
2010년 9월~현재: 우송대학교 테크노미디어융합학부 영상콘텐츠전공 교수
※관심분야 : 디지털방송 소프트웨어, 스마트TV방송UI/UX, 소프트웨어공학, 요구분석공학, N-스크린 서비스, 소프트웨어 교육