PROGRAMMING EDUCATION IN A WOMEN UNIVERSITY IN JAPAN: A CASE STUDY OF PERFORMANCE-BASED LEARNING IN LIBERAL ARTS

Natsuko Uchida
Ferris University Yokohama, Japan
uchida@ferris.ac.jp

Yasushi Kuno
University of Electro-Communications Tokyo, Japan
y-kuno@uec.ac.jp

Abstract

In Japan, Programming education for elementary school starts in 2020. In recent years, programming education at k-12 has been focused not only in Japan but also in other countries. However, not all stu-dents have opportunities to learn to program at Japanese colleges and universities. It is minimal for them. If college students have the chance to learn, they will only learn how to write code, not how to uti-lize what they have learned with actual software development. Due to this background, students have not been able to improve the ability to contribute to software development projects. Although IT-major departments provide advanced classes to learn those topics, non-IT-major students do not have such op-portunities. Authors consider that anyone, not just IT-major students, needs to learn to program before going out into the real world. Notably, we hope women challenge to learn programming. As a solution to this issue, the authors used limited materials and PBL (Project Based Learning) incorporating active learning. The characteristic of this course is to learn both "learning programming for the first time" and "learning important things in software development projects through practice" at the same time. Stu-dents can learn the important "abstraction" and "think elements needed to solve a problem" in learning programming by drawing diagrams to express their ideas. Besides, through project-based learning utiliz-ing LMS, they became capable to simultaneously learn the elements necessary for the real world (such as Division of roles, PM). In this paper, the authors discuss the design of the curriculum for program-ming education with results and effects.

A. INTRODUCTION

In recent years, with the development of digital society and the rapid progress of AI, the way of work in the society has changed. By now, such as mobile phones, robots, automated driving of cars, which peo-ples had dreamed about, have been realized and IT is becoming more and more familiar. According to a survey conducted by the Japanese Ministry of Economy, Trade, and Industry, about 590,000 IT person-nel are in demand by the year 2030 (METI 2016). However, the number of information science departments in Japan is not large, and it is impossible to satisfy the demand of IT human resources by graduates of these specialized departments. Thus, the development of IT human resources to cope with it has become one of the social issues.

In order to supplement these, it is necessary not only to develop human resources in various fields in "Ri-kei (Engineering and Science)" colleges but also to promote IT education widely among students in "Bun-kei (Humanities and Social Science)" colleges (Hagiya, 2015). Looking at work after graduation, the future society requires not only the ability to use the software but also the knowledge to engage in software development projects, even if the work is not directly related to IT. Thus, as basic education in university, it is necessary to learn to program and to acquire such knowledge on IT regardless of the ma-jor of "Bun-kei" or “Ri-kei.” Besides, the percentage of women in this field is low, and the IEEE activi-ties organize Women in Engineering (WIE)[1] to support women's activity.

The Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) has revised K-12 curriculum guidelines so that programming education in elementary schools will start in the year 2020. And the curriculum in junior high schools and senior high schools is also going to change accord-ingly. However, the IT curriculum in higher education is up to each university’s policies and are not im-plemented in the same way as curricula changes in primary and secondary education.

Based on these a background, the authors consider it desirable to expand opportunities for learning com-puter science. Therefore, to do this, it is necessary to develop a curriculum that can be taught well even in “Bun-kei” university or non-IT significant courses. Authors’ idea is that, given the post-hiring activi-ties, what should be learned through programming learning are not only to write codes but also about essential elements in a system development cycle (such as requirement specifications, involvement in projects). The reason is that in software development, it is necessary to accurately judge what type of system is to be created before proceeding with construction. To overcome these problems and realize both “learning programming for the first time” and “learning important things in software development projects through practice,” the authors designed a new lesson curriculum that combines these two. Also, instead of learning according to the textbook, the authors utilized visual representation using figures (i.e., line, circle, triangle, rectangle, and so on) and students’ thought on their own. This idea allows to limit the various processes that impede programming learning, and the authors could lower the solid walls that beginners hit.

In this paper, the authors discuss the principles and design of the new curriculum, along with experi-ences in women university. In Chapter 2, the authors analyze the situation in higher education in Japan. In Chapter 3 and Chapter 4, the authors propose the course design and class experiences. In Chapter 5 and Chapter 6, the authors present discussions and conclusion.

2. SITUATION FOR HIGHER EDUCATION IN JAPAN

2.1. Information Education and Programming Education in Colleges

One of the challenges is to realize information education and programming education in colleges by it-self. As mentioned above, in Japan, K-12 education is provided under the curriculum guidelines estab-lished by MEXT. However, in higher education, such guidelines do not exist, and college operates by their own curriculum.

According to a survey of 750 universities in Japan by MEXT conducted in November and December 2016 (Takahashi, 2017), there is a large number of undergraduate departments that do not teach “Algo-rithm and programming.” Instead, most of them only taught subjects like “computer literacy” and “in-formation ethics and security.”

Many students learn information literacy as an elective subject, and these courses are how to operate specific applications (ex. MS-Office). Therefore, most non-IT students do not have basic knowledge of computer science (CS) or programming. Consequently, they will enter the real world without under-standing that knowledge.

Despite this situation, on March 29, 2019, the government in Japan requested all universities to provide primary education of artificial intelligence (AI) to develop AI human resources. Authors agree that eve-ryone should have a basic knowledge of programming to adapt to the AI era. Furthermore, the authors believe it essential to develop a curriculum in which everyone can learn programming. However, the traditional style of programming education heavily relied on textbooks and lectures and postponed prac-tices to write their own code. Such a method will give many hurdles to learners, as they must obtain much knowledge; many learners will overflow or drop out. Even if everything is packed and learned, it is challenging for a novice or non-IT students to write a program that uses all functions. Thus, the au-thors propose to carefully select and limit what is necessary and encouraging them to write their code from the beginning.

The authors think that learning to the programming is not only the writing of the program but also the requirements specification and involvement in the project are essential elements from the viewpoint of post-employment activities. Because in software development, it is necessary to make appropriate decisions on what kind of system to build, to proceed with development according to schedule, and to complete the order.

Students majoring in information systems are allowed to learn and practice systems development think-ing through various subjects after an introductory programming course. However, students of non-IT major will not be able to do serious software development from an entry-level level and will not have the opportunity to learn the key points above.

The authors surveyed papers published by the Information Processing Society of Japan since 2000 on programming education in liberal arts universities and found that there were 11 peer-reviewed papers, study groups, and symposium presentations. These fall into two categories: three of the article were of educational support systems (Kawamura et al. 2001, Kato et al. 2015, Kato et al. 2017) and seven of the papers were of actual examples conducted in information education subjects (Yoshida T. 2008, Cho 2009, Matsumoto and Yoshida T. 2011, Ishiyama and Kusunoki 2012, Matsuzawa and Sakai 2013, Ishikawa 2003, Yoshida A. et al. 2015). However, they mainly focus on tools and languages, and few articles described methods for programming education itself.

Kawamura presents several points to keep in mind and examples of education using JavaScript in gen-eral information education at liberal arts universities (Kawamura 2011). Nishida et al. describe the ef-fects of courseware on learning, and in it, he mentions the impact of using graphics drawing type teach-ing materials (Nishida et al. 2017). Yoshida T. also suggests the introduction of programming in 4 of 15 lessons using PEN (Yoshida T. 2008). Yoshida A. et al. describe how to teach minimum knowledge in 3 learning phases (Yoshida A. et al. 2015). The critical point is how well teachers design a course for limited class hours.

2.2. Active Learning and PBL

Another recent trend in educational domain is Active Learning (AL). Today's central idea of learning is not passively accepting knowledge, but actively incorporate the experience into subjects, and it is done by associating the existing skills of the learner with the new one acquired (National Research Council and Division of Behavioral and Social Sciences and Education 1999, Papert 1980). The way of learning incorporates this idea is called active learning. Active learning is not learning passively by listening to the teacher's explanation; instead learners should take various kind of initiative with their problems.
In the report of the Japanese Central Council for Education "Improving Connectivity between Primary and Secondary Education and Higher Education" in 1999, was focused on career education in universi-ties. Following that, MEXT recommended AL at universities in August 2012. MEXT's 2016 college entrance reform addressed the limitations of traditional (knowledge-based) learning methods and sug-gested that future school education should focus on AL (NEXT 2016). In response to this, the following curriculum guidelines of primary school, junior high school, and senior high school publicly announced by MEXT from 2017 to 2018 emphasizes AL.

There are many options for how to proceed in active learning. One of them is PBL (Problem/Project-Based Learning, problem-solving type learning). PBL is initiated by American medical science educa-tion in the 1960s and 1970s with the Problem-Based as the axis, and in engineering education, there are many things done under the name of Project-Based (Sometimes "Problem" and "Project" are not distin-guished).

Since its inception in medical science and engineering, PBL has been practiced not only in these fields but also in many other disciplines, and its purposes vary, including the development of competencies, the development of problem-solving skills, and regional development/ improvement (Nakayama 2013, Ikemoto et al. 2014, Uchida Y. 2015).

Concerning software development education, it is easy to include PBL as the framework. The reason is that there is an apparent "PROBLEM" toward which software is created, and group development is es-sential depending on the size of the software. MEXT has implemented an enPiT (http://www.enpit.jp/) project, which aims to develop human resources to add knowledge and develop practical abilities to dis-cover and solve problems in 4 fields (cloud computing, security, embedded systems, business applica-tions) through the cooperation of multiple organizations.

Education with AL/PBL is challenging in various respect. Firstly, these teaching methods take much time, and some teachers and students are burdened and exhausted. Therefore, it is challenging to design a course in which everyone can work efficiently and meaningfully with PBL framework.

2.3. Connections to Society

In recent years, the abilities required in society include the following.

- Ability to solve difficult problems that might have multiple solutions or the existence of a solution is unclear
- Move autonomously
- Achieve results with a team of multiple people

The authors believe that AL at universities requires not only active learning but also a deeper level of education. Is learning genuinely equal to learning a lot? In the next AI era, in addition to these abilities, programming skills are required in particular, however, how can you introduce to learn them to all stu-dents?

Even before the term PBL became standard, in the college of information technology, practical training classes for software development have been conducted on a project basis. Also, in recent years, there have emerged practices that provide software development education with an awareness of the charac-teristics of PBL. (Matsuzawa and Oiwa 2007, Matsuzawam et al. 2008, Inoue and Kaneda 2008). How-ever, at "Bun-kei" colleges, there are tight opportunities to learn due to time constraints or limitation with teachers, misunderstanding of the need for information technology skills, and so on. The authors consider that it is possible to facilitate the connection to society by combining the two, programming and PBL above, and investigating comprehensively.

In the following chapter, the authors propose a PBL curriculum that overcomes the problems described so far and learns programming effectively in a limited time.

3. COURSE DESIGN

3.1. Purpose of the Curriculum

The purpose of the curriculum is to enable learning what is useful after students start to work. Therefore, the authors have targeted the curriculum both toward “learning programming for the first time” and “learning important things in software development projects through practice.” To achieve these goals, authors first considered what abilities are required explicitly in both “software development project” and “programming.”

Komaya says that PBL is effective in raising practical, independent, and problem-solving skills, and de-veloping human resources required by industry. Additionally, this educational method is expected to fill the gap between a company’s request and college education. The authors focused on what the industry (IT company) wanted to emphasize differently compared to educational institutions from the data in "IT human resources white paper 2013" (IPA IT Human Resource Development Headquarters 2013). The authors summarize the top items which the industries and educational institutions emphasize in Table 1.

Table 1 Comparison of Important Education Items

Rank	Item Name	Company	Educational Institutions
1	Communication Skill	61.2	59.4
2	Problem Solving Ability	42.6	29.3
3	Writing Skill	30.7	20.1
4	Teamwork/Cooperativeness	25.5	24.7
5	Programming Skill	22.2	19.2
6	Request Analysis	12.1	3.3
7	Project Management	11.7	8.8

Units : %

Table 1 summarizes responses from 564 companies and 239 educational institutions. The first to fourth in the list are general-purpose skills, what both institutions rates high. However, in comparison to the fifth and below place, requirements analysis and project management which rank sixth and seventh in the industry, just scores far lower in educational institutions. These data have shown that corresponds to the divergence of industry-academic awareness pointed out by Komaya (Komaya 2009). Considering social needs, the authors have selected seven items listed in Table 1 and additionally “software develop-ment process” skill. Note that four out of those eight items are related to software development.

As for the ability related to “programming,” it is unsatisfactory if “Simply learning the grammar of the language” or “Just copy the example code and run it” is the actual learning contents; in addition to these, learners need knowledge of software development. However, in many of the colleges (especially in “Bun-kei” majors), students tend to have low interests in those technical topics, and lesson times are short. Therefore, it is challenging to have students experience full-fledged system development.

Thus, the authors have decided to allocate minimum hours to programming languages and algorithms and to assign remaining hours to “making things” within Project-Based setups. Based on the above policies, the authors set the following four goals:
(1) Programming:
Programming should realize what you thought as a code, not just the experience of driving an ex-ample
(2) Project management (PM):
Students learn how to proceed with projects through group work. Besides, they should gain skills in effective communication, hands-on reporting, and presentation
(3) Requirement Analysis:

For purposeful development, students should experience analysis on “what is important” and “what type of function is implemented.”
(4) IT skills:
Students should utilize various IT tools for editing and debugging code, sharing information for group work, creating a deliverable, etc.

3.2. Curriculum Structure

Goals (2) through (4) mentioned in the previous section will not be cultivated through individual learn-ing but nourished by group work. However, since novice students do not know programming well, it is challenging to write codes in groups from the first lesson. Therefore, the authors allocated traditional lec-ture-practice style programming lessons in the first half of the classes. Afterward, the authors have con-figured students to multiple teams and implement the group work of software development in the sec-ond half (Uchida N. et al. 2017, Uchida N. 2018).

By doing this, even students with no prior programming experiences and no software development’s backgrounds could help each other in group work and could experience development tasks. Addition-ally, those types of knowledge and skills acquired through repetitive practice could be learned experien-tially while repeatedly practicing in the project.

Table 2 shows the course contents for 15 class hours (1 class hour corresponds to 90 minutes).

Table 2 Course Contents

Lesson	Contents
1	Introduction
2	Selection and Repetition
3	Control Structure and Arrays
4	Procedure and Abstraction
5	2-Dimensional Arrays and Images
6	Summary of programming part
7	Group creation / Assignment and goal setting
8	Brainstorming and Specifications
9	Design / Decide share / Production
10	Prototype / Intermediate document creation
11	Intermediate presentation / Review of specifications
12	Production / Unit test
13	Code review
14	Production / Integration test
15	Final presentation (summary)

3.3. Programming Part

This course is for non-IT course students who are studying programming for the first time. Students need to develop the skill to “write code as they think” which will be required in the latter part of the class. The authors used the guidelines of Kuno for learning programming (Kuno 2016).

The authors think that the followings should be avoided in programming education.

- Require to "just memorize" lots of knowledge written in textbooks.
- Repeated practices with drills and other materials targeted toward memorization, e.g., "what code pattern appear most in the examination" and so on.
- In testing, asking students to solve many problems in a short time with "remembered" way.
- Instead, the authors think that the following points should be pursued in effective programming education.
- Students should understand "minimum rules for programming" and make those rules repeat-edly in various way. Thereby they should practice "how to think like a programmer."
- Students should choose the problem they solve according to their skills and interests, which results in better learnings experiences and better motivation.
- Students should understand that there are large varieties in writing "correct" programs, and there is no "single correct solution."

Moreover, the conventional textbook has the following undesirable features.

- Rules (syntax and functionalities) of the programming language are explained one by one in the textbook, and students are expected to learn (or memorize) them one by one.
- Many of the problems are not about "writing a program," but about recalling rote knowledge of the explained contents.
- There are only a few examples of the program with through (or detailed) explanations, suggesting bare memorization of those examples.
- Alternatively, the authors have adapted the text with following features.
- Minimum rule of thumb is explained, and students are encouraged to apply those rule through various exercises.
- Many numbers of exercises are provided with various difficulty; students choose their exercise to solve according to their levels and interests.
- Many topics and example code are provided, and students choose whichever topic to attract their interests.

Each lesson consists of "Explanation → Exercise → Assignment in the classroom (Assignment A) → Homework until the next lesson (Assignment B)." The authors prepared exercises from straight forward ones (Students can submit if they modify the example a little) to sophisticated ones. Students choose a problem according to their level from among those exercises. In this way, each student can obtain their learning according to their level, and it is possible to write "one's code" instead of "copying and running the example or someone else's answer."

As mentioned earlier, when learning programming, novices must learn much from the first lesson. However, there are too much and challenging things for them. Hence the authors designed the limited course contents which utilizes code for drawing diagrams (round, triangle, squares, and line). Using these, students learn the necessary items of programming (selection, repetition, function, arrays) just as in standard methods.

In this part, although the contents are minimal, the goal is to write one's code and to acquire the mini-mum programming skills necessary for the project.

3.4. Project Part

In the second half (starting from the 7th class), four to five students form a group. The purpose of this part is to acquire practical skills of programming by experiencing the project and gaining the ability to operate the project.

Based on the programming skills acquired in the first half, they set up for more practical tasks and experience request analysis (corresponding to target (3) in section 3.1) in the process of deciding what type of work to make. Although it is difficult to learn everything in one project, in this curriculum, we have prepared an environment in which practical learning occurs, therefore that the project can be completed within the planned timeline, and with the consciousness of project management.
As a failure of PBL, several prior research (Inoue and Kaneda 2008, Uchida Y. 2015) mentioned the following two.

- One person works in group work, and the other members do nothing
- The quality of output varies depending on the ability and skill of members

For these problems, the authors thought it would be beneficial to clarify the role within the group and share information. As one of the tools for that purpose, the authors use a worksheet to record the sharing and progress each time. Another idea is to set up multiple opportunities for all the students to share how each group is working, and to have a place to exchange ideas among the groups, not just in the group discussion. (Corresponds to the 11th week's interim announcement and the 13th week's code review in Table 1).

In this course design, as an essential assignment task is identical throughout the groups (only differ with what to make as the output), all students of all teams are sharing the same goal. Therefore, they could discuss in the same viewpoint. In each group, spirals of "production (creation)" → "sharing" → "reflec-tion" are formed, and in the classroom, the flow of "sharing" → "empathy" → "competition" is created. In this design, students could brush up their programming skills through compensation within each group, and those skills could further be expanded through the sharing of information within and among the groups. Also, students' motivation could be kept high due to the awareness of competition among the groups.

There are four expectations for the learning effects that these processes (to organize one's thoughts — make a trial — complete one's work) will provide.

- Supplementing programming skills that were insufficient in the first half.
- Expand your scope by sharing information among groups.
- Motivates you by creating a sense of competition among groups.
- Experientially learn how to communicate accurately

[1] CLASSES EXPERIENCES

1.1. Class Outline

Here the authors describe the class experiences on women's school, Ferris University[2] during the sec-ond semester of the school year 2017 and 2018. The course was in the Faculty of Global and Intercul-tural Studies. It was an elective subject for first through fourth-grade student and was also open to other undergraduate departments (Faculty of letters and College of Music). Classes were conducted once a week in 90 minutes. As described before, introduction to programming occupies week 1 through 6, and the remaining weeks project activities (Table 1).

The authors used Ruby programming language due to simple syntax and suitable execution environ-ment (read-eval-print loop with automatic output). As the authors were not able to hire an assistant in this course, the authors recommended pair programming as a way to get by with only one teacher (one of the authors).

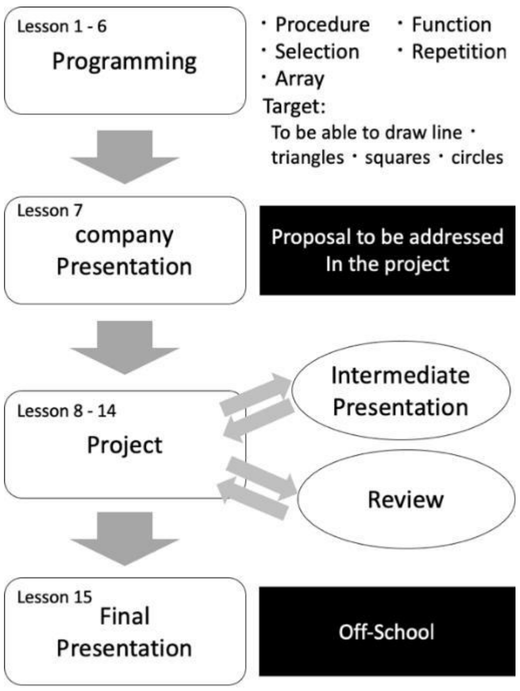


Figure 1
Overall Flow of the Course

The environment of the classroom is Windows 10, and it is a simple setup just installing Ruby language and ImageMagick (image display software). As an editor, initially, the authors used the Windows stand-ard Notepad. The authors changed it to Atom as of issues with Japanese code and consideration for Mac users. The goal of the programming part was to be able to draw figures (lines, triangles, squares, circles) neces-sary for the project part, focusing on basic control structures such as selection and repetition. Using a[2] Ferris University: Ferris was founded according to the spirit of Christianity for the purpose of educating women. The first article in the school regulations states that “This school is an academic research and educational institution, its educa-tional activities conducted in the spirit of Christianity, providing women with a high level of education and undertaking specialized scholarly research, with a love of truth and peace, and aiming to cultivate people who can contribute to the welfare of humanity.” (https://www.ferris.ac.jp) graphic has the advantage of being able to output an execution result visually. In the project part, every group consisted of four members. The authors have determined the grouping based on the submission, individual students’ understanding level (as observed in previous lessons), and intermediate question-naire. The intermediate questionnaire consisted of self-evaluating questions about programming under-standing, group work skills, and other generic skills.

In the 11th and 13th week, the authors have set up an interim presentation by each group, shared pro-posals, and progress of each group. Additionally, in the 15th (final) week, the lesson was held in the conference room of the cooperating company. Figure 1 shows a schematic representation of the course.

4.2. Submitted Issue

Many of the precedent cases of PBL used real-world system development as the class material. How-ever, it is too much of a challenge to tackle such issues in the programming subjects for beginners. Therefore, the authors have asked the cooperating company to provide a practical but simplistic task for the student projects.

The authors strived hard to assimilate real-world processes and devised a process not so difficult for stu-dents. There are many examples of programming using graphics. However, there are no examples of connecting the task to manufacturing. Many of the cases are the ones that individuals tackle. The authors thought that students could work on drawing in a process similar to system development, with themes and materials they were interested in their own. Additionally, the authors have asked the cooperating company to provide a straightforward task for students' projects. The specific assignment was to create a design toward some specific target (each group decides their goal), and a GIF animation for advertis-ing.

4.2.1. Case in 2017

The assignment was to design a nail sticker that a female student is interested in it. Students made a code for a nail sticker toward some specific target (each group decides their target of person), and a GIF ani-mation for advertisement. Instead of working individually on each task, students share design and work together among team members. Figure 2 shows part of the created nail sticker. Figure 3 shows the de-velopment of part of the animation.

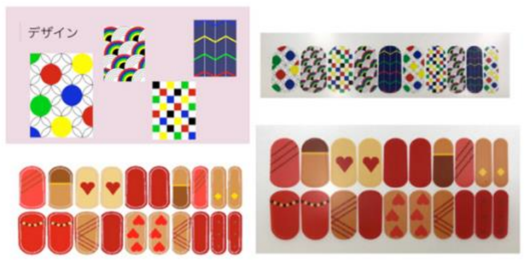


Figure 2 Outcome: Nail Sticker

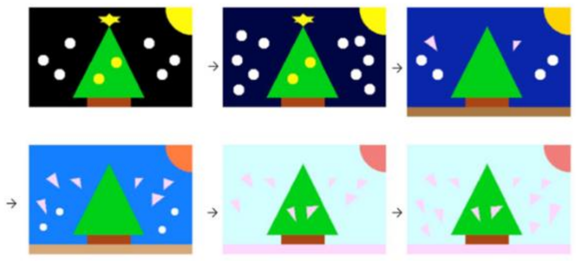


Figure 3 Outcome: fig Animation

4.2.2. Case in 2018

Making use of the reflection in 2017, the authors changed the way the programming part goes and the text. The authors decided to draw pictures aggressively from the first lesson, excluding mathematical elements as much as possible. The author also added the appearance of toilet paper appearance as an in-termediate subject (end of programming part). Figure 4 is a part of the work. Figure 4 is a part of the works.

In the project part, the authors changed the nail sticker into a clear file folder and worked by the same process as in 2017. The theme was to combine three keywords: Yokohama, ZOORASIA (zoo), and Rose. Figure 5 shows the design of the created clear file folders.

YOKOHAMA
SOTETSU

Figure 4 Outcome: Toilet Paper Appearance Figure 5 Outcome: Clear File Folder

5. DISCUSSION

5.1. Learning Programming for the First Time

In this research, the author focused on the connection between knowledge learned at university and the real-world use and examined how to introduce programming education meaningfully making use of a limited environment. First, we discuss "learning programming for the first time." As in the case of other researches (Yoshida, T. 2008, Matsumoto and Yoshida 2011, Nishida et al. 2017), the authors used graphic drawings as teaching materials. Besides, the authors prepared teaching materials with limited content and project-based settings in this course. The purpose is to enable students to confirm their thoughts through programming. Using drawing, students can easily check design, cod-ing, and result of outputs. An example is shown in Figure 6. These processes foster the logical thinking skills needed to get the computer to do the intended processing.

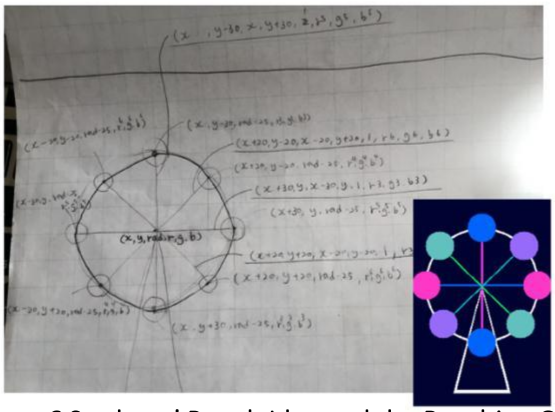
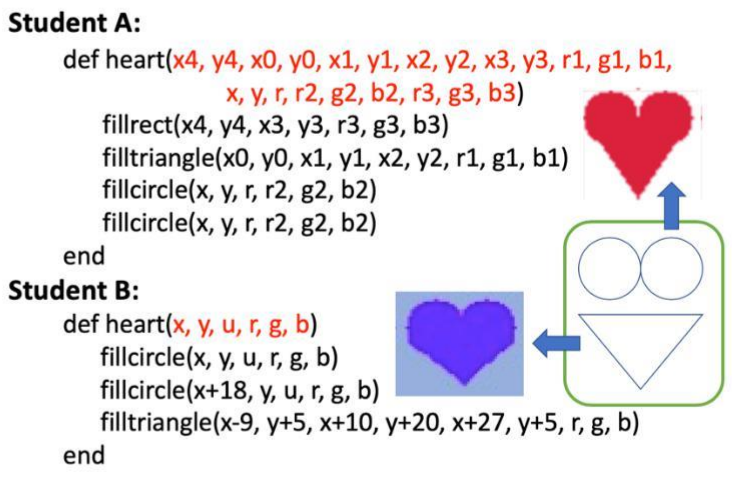


Figure 6 Students' Rough Idea and the Resulting Output

Modeling and the process of selection or repetition are important items when learning programming at first. When learning programming for the first time, it is difficult to write a full-fledged software pro-gram because there is a lot to learn. However, figures are useful because they are easy to grasp visually.

On the other hand, the problem with programming learning with graphics is that learner merely repeat plotting the drawing methods without using selection or repetition (Yoshida, T. 2008). The same situation occurred in authors case. Many of these problems can occur when drawing simple diagrams. Also, the same thing happens when drawing too complicated. The authors think the reason is in the teaching materials. Because the goal in the drawing is simple, most of them concentrated on the arrange-ment of the parts (such as lines, circles, triangles.) In both cases, the authors consider that the students are focused on the output results and ignoring intermediate processes. Therefore, it was difficult for them to use a kind of selection or repetition. This problem was solved not only by writing a program for homework but also by imposing assignments that students were interested in and making things.

Next, when writing a program, the principal thing is abstraction and parameters. Figure 7 is example of the method for drawing the heart shape of two students.



```
Student A:
    def heart(x4, y4, x0, y0, x1, y1, x2, y2, x3, y3, r1, g1, b1,
                  x, y, r, r2, g2, b2, r3, g3, b3)
        fillrect(x4, y4, x3, y3, r3, g3, b3)
        filltriangle(x0, y0, x1, y1, x2, y2, r1, g1, b1)
        fillcircle(x, y, r, r2, g2, b2)
        fillcircle(x, y, r, r2, g2, b2)
    end
Student B:
    def heart(x, y, u, r, g, b)
        fillcircle(x, y, u, r, g, b)
        fillcircle(x+18, y, u, r, g, b)
        filltriangle(x-9, y+5, x+10, y+20, x+27, y+5, r, g, b)
    end
```

Figure 7 Example of the Method for Drawing the Heart Shape

Both students used two circles and one triangle. However, student A was confused by not clearly under-standing the procedure call and the arguments. As a result, the program was not completed. Student B made a method that consists of a small number of parameters by modeling the relationship between a circle and a triangle well.

Making the method is one of a significant factor when learning programming. However, students tend to plot only the parts without thinking well; like a student A. The authors shared this example in class, and then students understood well about the algorithm and parameters. The authors consider that sharing their common problems increased student interest or motivation and promoted learning effectiveness.

When learning to program, there is often a tendency to learn a lot and tackle complex tasks. The author suggests an environment where students learn deeply with simple materials like this example, rather than addressing complex issues.

5.2. Learning by Project through Practice

The authors discuss the viewpoint of "learning important things in software development projects through practice."

In this trial, the authors recommended pair programming as an introduction to the project part. Matsuura et al. and Hirai et al. state that pair programming among novices does not necessarily lead to good re-sults (Matsuura et al. 2003, Hirai et al. 2012). However, the authors got the opposite effect form stu-dents' feedback of pair programming in the programming part. There are representative comments from two students derived from feedback in the class.

- "Today, the first-grader who is paired with me succeed to run the program well for the first time. I saw that she could get to grasping a knack."
- "Since this class was pair work, I could find solutions and mistakes while teaching the pairs and thinking together when I got an error. With this experience, I feel that I could get to gain the skill."

In the class, the authors have taught students about the importance of not only learning programming languages but also learning how to use the programming language at software development. This course is an introductory-level activity at the Liberal Arts University. Therefore, the quality differs from that of an IT major; however, these knowledges or thinking skills become understandable by simplifying the teaching materials, and students could solidify the basics knowledge.

There are many programming educations courses or manufacturing PBL classes. However, it is difficult to take enough time due to limited on class time. Therefore, once students complete their output, they are not easy to rethink or reinvent it. However, the authors' limited curriculum can incorporate that time in intermediate presentations and code reviews in project parts. Figure 8 shows the student's prac-tice. She draws the tower.

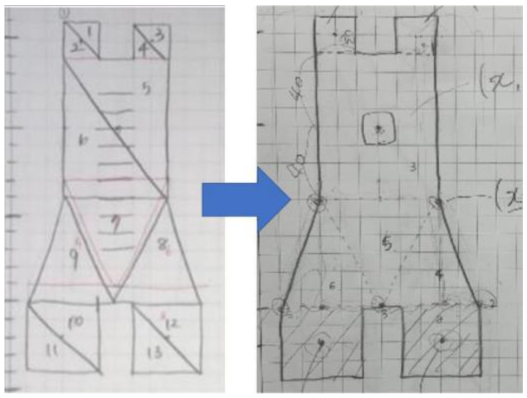


Figure 8 Rethinking the Configuration

At first, she thought to use only triangles. She thought this way is simple. However, after a code review, she noticed that the use of triangles and rectangles simplified the structure. This case is one of the simple examples, but the combination of simple shapes has made it clear for students to think visually. In addi-tion, group discussions made it easier to secure time for rethinking or reinventing.

On the other hand, there are issues related to the behavior of each student in group work. As shown in the previous research, the disadvantage of PBL is the difference in the amount of activity for task assignment and the quality of output. In primary and secondary education, students study the same content (goal). However, when they work in society, they share work and do different jobs from each other; therefore, they will not do the same work at the same time. For such occasions, it is essential to be conscious of what kind of work they have in class, how to share it, and how to complete it by the deadline. The authors used the worksheet for clarifying the students divide the role of the task and work in every lesson.

6. CONCLUSIONS

In this paper, the authors described the design of a new course curriculum based on PBL which achieves both "learning programming for the first time" and "learning important things in software development projects through practice" with the minimum number of hours. Moreover, the authors discussed the case study of restricted content in the introduction of programming education at a women University in lib-eral arts.

Due to the limited teaching material, students did not gain expert-level knowledge. They have only ac-quired entry-level experience, but this proposed curriculum was very successful as an en-try-level sub-ject.

Many "bun-kei" university classes are centered on traditional lecture-style and have little practical or ex-periences toward learning. In such an environment, it is difficult to expand the situation to learn actual (usable) programming. Therefore, the hurdle for students to learn programming is high. Our curriculum design might lower those hurdles.

In project-based learning, the burden on teachers is also an issue. Further, a teaching assistant is a "must" on programming practice classes, which is difficult in "bun-kei" university. In such cases, the burden on teachers well become doubled or tripled. However, in our curriculum design, after the projects are suc-cessfully launched, students start to learn by themselves / among themselves toward their goals, lower-ing the burden noted above.

A case study in 2018, students learned the important "abstraction" and "think elements needed to solve a problem" in learning programming by drawing diagrams to express their ideas.

From the practice of this study, the authors got good results on the learning effect of utilizing a real pro-ject even on a small scale. The learning effect of students when making products is high. The authors guess that the appropriate goal setting realized engagement (MEXT 2015), which improves the motiva-tion of the students and also the satisfaction after the class.

All in all, our curriculum design of "programming and project in one subject" could easily be transferred to many environments, and will contribute to better class experiences both for students and for teachers.

REFERENCES

1. Cho, S. (2009). Learning Recursive Programs for non - CS students [in Japanese]. SIG Technical Re-ports, 3(2009-CE-99), pp. 1-8.
2. Hagiya, M. (2015). Invited Papers Defining Informatics across Bun-kei and Ri-kei. Information Pro-cessing Society of Japan, Vol. 23, No. 4, pp. 525-530.
3. Ikemoto, Y., Suzuki, N., Kondo, A., & Yamamoto, K. (2014). University education, Project-Based Learning, PBL, competency, image media, specialized compulsory course [in Japanese]. Shikoku Uni-versity.
4. Inoue, A., & Kaneda, S. (2008). A PBL Approach using Real World Application Development be-tween University and Local Government [in Japanese]. Information Processing Society of Japan. IPA, IT Human Resource Development Headquarters. (2013). IT Human Resource White Paper 2013 [in Japanese]. Information-technology Promotion Agency, Japan. Retrieved from https://www.ipa.go.jp/files/000027245.pdf
5. Ishikawa, T. (2003) Object-Oriented Programming Education Practice for Liberal Arts College Students [in Japanese]. Information Education Symposium 2003 Proceedings, pp. 77-82
6. Kato, Y., Matsuzawa, Y. and Sakai, S. (2015). Analysis of Interaction in Novice Collaborative Pro-gramming [in Japanese]. Information Education Symposium 2015 Proceedings, pp. 177-184.
7. Kato, Y., Matsuzawa, Y. and Sakai S. (2017). CheCoPro: A Software Promoting Collaborative Pro-gramming for Novices through Collaborative Knowledge Creation Approach [in Japanese]. IPSJ Trans-actions on Computers and Education, vol. 3, no. 2, pp. 28–40.
8. Kawamura, k. (2011). About the way of Programming Education in General Education [in Japanese], SIG Technical Reports,16(2011-CE-108), pp. 1-8.
9. Kamaura, K., Hishina, M., and Tokuoka, K. (2001). Design and Development of Programming Educa-tion Support System JPADet (Japanese PAD Editor and in Terpreter) for Liberal Arts Students [in Japa-nese], SIG Technical Reports, 122(2001-CE-062), pp.9-16.
10. Komaya, S. (2009). What is the impact of PBL on education [in Japanese]. Summer Symposium in Saga 2009. The Information Processing Society of Japan.

11. Kuno, Y. (2016). What purpose do you learn programming? And how? [in Japanese]. Retrieved from http://lecture.ecc.u-tokyo.ac.jp/ckuno/is16/sprosym-2016.pdf
12. Matsumoto, K. and Yoshida, Y. (2011). Programming Education for Liberal Arts College Students with PEN—Proposal of New Materials for Teaching Loop Structure— [in Japanese]. SIG Technical Re-ports, 2011-CE-109, Vol.7, pp.1-7.
13. Matsuzawa Y. and Ohiwa H. (2007). A Result of Trial Education for Software Engineers through Uni-versity-Industry Collaboration and Project-based Learning [in Japanese]. Journal of Information Pro-cessing Society of Japan. vol. 48, no. 8, pp. 2767–2780.
14. METI. (2016). Summarized survey results on latest trends and estimates for IT personnel [in Japanese], Ministry of Economy, Trade and Industry, https://www.meti.go.jp/press/2016/06/20160610002/20160610002.html
15. MEXT. (2015). Curriculum Planning Special Committee [in Japanese]. Ministry of Education, Culture, Sports, Science and Technology, Retrieved from http://www.mext.go.jp/compo-nent/b_menu/shingi/toushin/__icsFiles/afieldfile/2015/09/24/1361110_2_5.pdf
16. MEXT. (2016). Final Report [in Japanese]. Council for Reform on the System of Articulation of High Schools and Universities, Ministry of Education, Culture, Sports, Science and Technology. Retrieved from http://www.meti.go.jp/policy/it_policy/jinzai/27FY_report.html
17. Nakayama, R. (2013). Facilitating active learning by the introduction of problem-based / project-based learning (PBL) [in Japanese]. Japan: Center for 21st Century Education, Hirosaki University. National Research Council and Division of Behaivioral and Social Sciences and Education. (1999). How People Learn. Washington D.C.: National Academy Press.
18. Nishida, T., Harada, A., Nakanishi, M. and Matsuura, T. (2017). Comparison between Two Types of Courseware for Introductory Programming [in Japanese]. IPSJ Transactions on Computers and Educa-tion, vol. 3, no. 1, pp. 26-35.
19. Papert, S. (1980). Mindstorms: Children, Computer, and Powerful Ideas. Basic Book.
20. Takahashi, N. (2017). National Survey of Japanese Universities on IT Education: Preliminary Analysis of General IT Education [in Japanese]. Information Processing Society of Japan.
21. Uchida, N., Montenegro, C., Kuno, Y., & Kakehi, K. (2017, October 5 - 7). Importance of Course Con-tent and Classroom Design in Project-Based Learning in Programming Courses –Case Study of a Lib-eral Arts College, Ferris University in Japan –. Philippine eLearning Society (PeLS).
22. Uchida, N. (2018, February 15). "Peta-gogy" for Future: Novice Programming Courses and PBL [in Japanese]. IPSJ magazine, 3(59), pp.268 - 271.
23. Uchida, Y. (2015). Practice of Creative Education to Learn the Problem-Solving Capabilities [in Japa-nese]. Japanese Society for Engineering Education.
24. Yoshida, A., Kishi, N. and Abe, K. (2015). A Report on Teaching "K-12 Computer Education" to Col-lege Students [in Japanese]. SIG Technical Reports. 2015-CE-129, pp.1-8.
25. Yoshida, T. (2008). Programming Education for Liberal Arts College Students with PEN [in Japanese].
26. SIG Technical Reports, 64(2008-CE-095), pp. 71-78.