Information Engineering Express
International Institute of Applied Informatics
2019, Vol. 5, No. 1, 73 – 85

Factor Analysis to Examine the Effect and Expectation of Reading Simple Source Codes for Improving Programming Skill

Shimpei Matsumoto [*], Tomoko Kashima [†],
Shuichi Yamagishi [‡]

Abstract

The authors have developed a programming learning support system based on source code reading whose primary target is novice programming learners. This system trains its learners by giving questions requiring to answer a variable's value after the execution of a presented source code. This paper shows the details of the developed system and the examination result on the effect and the expectation of reading simple source codes based on the real operation to a basic programming lecture. After the learners had completed one semester of the instruction in programming by utilizing the developed system, we obtained learner's responses by a questionnaire and analyzed it with factor analysis. The analysis result unveiled the characteristic factors and these degrees.

Keywords: factor Analysis, learning support system, programming education, reading based programming learning

1 Introduction

Information engineering education at Japanese universities has faced an extremely important issue of how to support learners not good at programming. To efficiently educate novice programming learners like this, the instructors have to clarify the cause preventing the understanding of programming, and also develop a new teaching method appropriate for each learner's understanding degree. As far as the authors' survey, development of an instructional method for learners not good at programming is insufficient. One main cause of this could be that programming requires various skill such as logical thinking, language, imag-ination, ingenuity, and of course, mathematical ability. On the other hand, we are not still sure what and how much skill is necessary to master programming. Additionally, there is no method to indicate learner's skill level in each knowledge domain, so learner is also not sure what kind of skill should be trained to overcome the difficulty of programming. Therefore, to show the appropriate direction for learning, the

[*] Hiroshima Institute of Technology, Japan
[†] Kindai University, Japan
[‡] Hiroshima Institute of Technology, Japan

authors have been considering that rogramming lecturer needs to create learning materials after limiting skill category and also show the attainment target of each skill for learners. That is, it is necessary to clarify what kind of basic skill is trained by reading/writing a program code. Besides, programming lecturer also should provide opportunities enabling novice learners to touch source codes easily by providing some kind of tool such as a learning management system (LMS). Since LMS can not only support daily programming education but also effectively collect learning log data which will contribute to defining the understanding degree in each skill field, it would be effective for supporting a learner not good at programming.

The authors have developed a programming learning support system based on source code reading, and the primary target of the developed system is novice programming learners. In our system, firstly the learner reads a source code given by the system and then answers a variable's value at the time when the presented source terminates. The aim of this learning is to give a lot of experiences to touch programs. Many novice programming learners who fear to program seem to be a lack of experiences in programming. Most of them tend to lose interest in programming because they cannot remove a program error themselves, and the drop of the interest eventually prevents the accumulation of the experiences. The authors have assumed that gaining experience of source code reading has a good influence for getting used to programming. The developed system can make a lot of chances for novice learners to experience programming because source code reading is simple to learn. The resistance to programming would be suppressed by getting used to programming, so source code reading based learning would contribute to improving programming skill. Additionally, reading source code is easier and more convenient to touch programs than writing, but is the fundamental skill strongly related to tracing, debugging, reviewing, and refactoring works. Therefore, source code reading based learning would be an effective learning way for novice programmers.

The objective of this paper is to utilize the developed system to an actual programming lecture and to examine an efficient way to use from the analysis result of factor analysis which is obtained by learner's responses. The lecture is for first-grade students majoring in Informatics, most of them are a beginner of programming, and used C language. This programming lecture continued one semester, and it consisted of 15 lectures. The developed system was utilized with examination style. Each question in all tests is a very simple program, i.e., it consists of about five instructions and all of which are assignment statements without complex calculation and control sequence. So, they are unreal programs but are just learning materials to require students to understand the processing rules of assignment statements and operators. We collected responses from a questionnaire, after the students had completed one semester of the instruction in programming, and analyzed the data. From the analysis result, it turned out that characteristic factors and these contribution rates were found. Concretely, experimental results suggested that the first factor is the evaluation for the developed system and the second factor is the evaluation for the contents of questions. By factor analysis, this paper was able to clarify the potential needs depending on student's skill level towards further learning support which would be the significant finding. Addi-tionally, based on the result of factor analysis, this paper discussed the future direction of learning support with source code reading-based learning.

2 Related Works

Previous studies on programming are roughly classified into three objectives [1][2]: analysis of student's learning history data and discovery of student's characteristic, proposal of a programming teaching method, and development of ICT-based learning support system [3]. So far, many researches on programming education have often focused on writing source code, but some of them have addressed reading source code [4]. Reading source code has also been considered to be an important ability for obtaining a deeper understanding of programming, but good reading strategies have not been known so much.

Earlier programming education studies on reading source code provide a lot of valuable knowledge about a characteristic of programming learners, teaching method, and program-ming learning support system. As a remarkable effort, a framework on reading source code was proposed [5]. Next, Arai et al. developed a learning support system for programming beginners that facilitates the process of learning by reading source code, and confirmed that the system was effective [6]. Tang developed a distributed, social code review tool designed for the specific constraints and goals of a programming course named "Caesar" [7]. Bus-jahn et al. took a further look into the role of reading source code in programming learning from the result of interviews with programming instructors using the miracle question, on the role of code reading and comprehension [8].

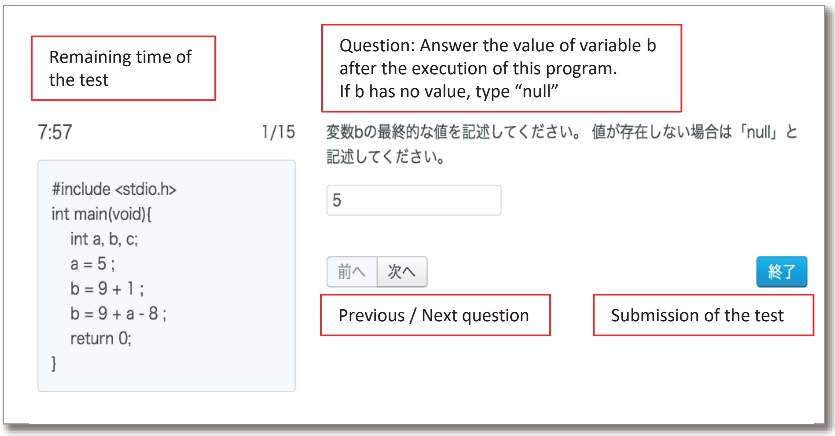
Reading and trace are closely related. As previous studies on tracing, Lister et al. showed that students who sketch a trace on paper had greater success on code 'reading' [9]. Cunningham et al. replicated this finding, and developed further categories of student sketching strategies [10].

3 Developed System

3.1 Source Code Reading

The developed system is for training the fundamental skill of C source code reading. Writing program is the particularly difficult task for learners not good at programming, so this system tries to reduce the psychological resistance to programming. The developed system aims to equip minimum knowledge, but necessary for programming without describing source code, and to rebuild his/her confidence to make program. The authors expect that writing skill of source code would be eventually improved because the psychological resistance will be reduced by his/her confidence of programming which is cultivated by reading many source codes. Additionally, reading many source codes will contribute to writing a readable code regardless of a particular context [11]. The developed system would also be useful as a method to grasp what and how much skill is insufficient for improving programming ability because it can collect each learner's response. The developed system can also enrich a programming lecture because it is available for self-study and mini-examination.

The question of the developed system requires learners to answer the proper value of a variable when the presented source code terminates. The developed system can provide the two-option question, the four-option question, and the free description question. The developed system can automatically generate a source code of C programming language used as a learning material, but a lecturer can freely edit the generated source code anytime such as after the automatic generation. The randomly generated source code consists of the series of meaningless procedure, so it does not have a specific purpose.


Remaining time of the test
Question: Answer the value of variable b after the execution of this program.
If b has no value, type "null"
7:57
1/15
前へ
次へ
終了
Previous / Next question
Submission of the test

```
#include <stdio.h>
int main(void){
    int a, b, c;
    a = 5 ;
    b = 9 + 1 ;
    b = 9 + a - 8 ;
    return 0;
}
```

変数bの最終的な値を記述してください。 値が存在しない場合は「null」と記述してください。

5

Figure 1: A screenshot of the developed system

This paper considers that reading source code would be effective to equip the basic of mental model for programming (the basic knowledge on programming language specification) even there is no certain meaning. With the developed system, the learner would also learn the basics of the processing flow, memory retention, calculation ability, and basic knowledge such as the assignment operator and the increment/decrement, which are the difficult concept for learners not good at programming. Besides, this learning would support to develop a habit of being conscious of program structure such as data dependence relationship [12].

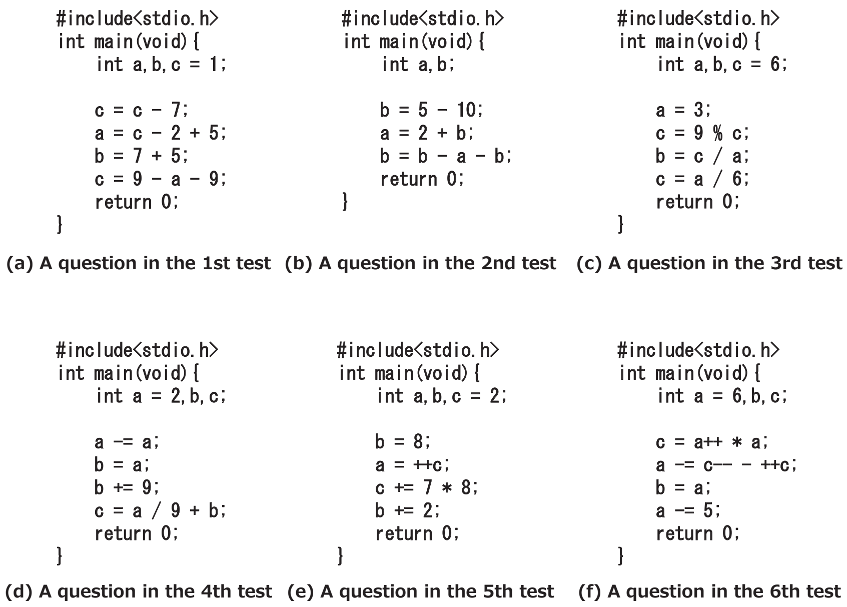
Usually, the total lines of a source code generated by the developed system are several dozens. Such the source codes are unrealistic because they are too short to process a certain meaning work, but this paper considers that reading source code described in a short sentence is important to get used to a large-scale practical program as a first step. In addition to these, recently since many software development fields have actively adopted programmer-centered software development approach with utilizing open source software, the demand of source code reading technique not depending on "comment" is now becoming greater and greater. Additionally, most of the time to program is said to be reading [13]. These backgrounds mentioned above are the main reasons that the authors focused on the source code reading blended programming instruction.

3.2 Specification of Implementation

The developed system is a web application. All functions of the developed system are available through the Internet as long as user-side device equips a web browser available the standard of HTML5. The developed system runs on an operating system with Linux kernel 3.16, and uses the following technologies: Apache 2.4.7, a web server program, to provide web service, PHP 5.5.9, a server-side scripting language, to implement the function, jQuery 1.7.2, a cross-platform JavaScript library, to perform dynamic UI, and MySQL 5.6.16, a database server program, to manage all system data. Fig. 1 shows an example of a screenshot of the developed system. The source code as shown in Fig. 1 is a question for a learner. As shown above subsection, some question types are available in the developed system, such as a free descriptive question, and a multiple choice question. This paper only addresses only the trial result of the free descriptive question. The learner has to put an integer number into a specified textbox from an input device. Before a programming

Table 1: Conditions to generate a source code

Term	S1	S2	S3	S4	S5	S6	S7	S8	S9
1	3	4	3	0.3	+,-,		*		
2	3	4	2	0.3	+,-		*		
3	3	4	3	0.3	+,-,*,/,%	+,-	*		
4	3	4	3	0.4	+,-,*,/,%	+,-	*		
5	3	4	2	0.4	+,-,*,/,%	+,-,*,/,%	*	*	
6	3	4	2	0.6	+,-,*,/,%	+,-,*,/,%	*	*	



```
#include<stdio.h>
int main(void){
    int a,b,c = 1;

    c = c - 7;
    a = c - 2 + 5;
    b = 7 + 5;
    c = 9 - a - 9;
    return 0;
}
```

(a) A question in the 1st test

```
#include<stdio.h>
int main(void){
    int a,b;

    b = 5 - 10;
    a = 2 + b;
    b = b - a - b;
    return 0;
}
```

(b) A question in the 2nd test

```
#include<stdio.h>
int main(void){
    int a,b,c = 6;

    a = 3;
    c = 9 % c;
    b = c / a;
    c = a / 6;
    return 0;
}
```

(c) A question in the 3rd test

```
#include<stdio.h>
int main(void){
    int a = 2,b,c;

    a -= a;
    b = a;
    b += 9;
    c = a / 9 + b;
    return 0;
}
```

(d) A question in the 4th test

```
#include<stdio.h>
int main(void){
    int a,b,c = 2;

    b = 8;
    a = ++c;
    c += 7 * 8;
    b += 2;
    return 0;
}
```

(e) A question in the 5th test

```
#include<stdio.h>
int main(void){
    int a = 6,b,c;

    c = a++ * a;
    a -= c-- - ++c;
    b = a;
    a -= 5;
    return 0;
}
```

(f) A question in the 6th test

Figure 2: Examples of questions in each term

class, the lecturer picks up a variable from a source code, and the variable is used as a question. There is another type of free descriptive question which requires learner to answer all variable values from a presented source code, but this capability is not mentioned in this paper.

All source codes used as questions consist of only sequential processing, do not include branch and repetition to give learners a firm foundation in programming. Besides, all source codes consist of only integer variables.

3.3 Automatic Generation of a Question

The developed system has the capability to automatically generate a source code used by a question. The instructor gives the following conditions to obtain a source code by the configuration form:

- S1: The number of variables: 1-9 variables are available, where variable names are alphabetical sequence, i.e., a, b, c, ...,i or array a[0], a[1], ..., a[9].

- S2: The number of statements (lines): 1-10 statements are available, where a closing curly bracket does not include the number.
- S3: The maximum number of variables calculated in a statement: 1-10 variables are available.
- S4: Appearance ratio of variables in a calculation: 0-100% is available. When con-structing an assignment statement, the variable or constant is selected according to the probability given here.
- S5: Use/disuse of operators: use/disuse of addition, subtraction, multiplication, dv-sion, and remainder operators are selectable. A source code consists of only simple assignment statement when all operators are disused.
- S6: Use/disuse of compound assignment operators: use/disuse of addition, subtrac-tion, multiplication, division, and remainder operators are selectable.
- S7: Use/disuse of increment/decrement.
- S8: Use/disuse of initialization.
- S9: Appearance ratio of a redundant statement (0-100%): e.g. “a=1; a=2;”. As the source code is constructed by giving each statement one by one in order from the top, a redundant statement is generated with the probability specified here.

4 Experiment and Result

This paper supported a programming lecture by utilizing the developed system. The lecture is to give the basic of C language for freshman students majoring in Informatics, most of them are a beginner of programming. This programming lecture continued one semester, and it consisted of 15 classes. The developed system was utilized with examination style in 6 classes among the 15 classes. The author provided 15 questions for each test, and each testing time is 10 minutes. The amount of knowledge in each test was related to the contents of the lecture according to the progress of this lecture. Concretely, in the early stage of this lecture, a test includes only simple calculation with addition and subtraction, on the other hand, in the closing stage, various operators and techniques, which has a possibility to create an awful statement such as “a += a++ - + +a;”, are given to generate a source code in the late stage. The detail of the conditions to generate a source code is shown as Table 1, and examples of the question in each test are shown in Fig.2. After each test, questions included in a test are available to practice as much as learner wants.

This paper obtained 107 student responses from a questionnaire, after the students had completed one semester of the instruction, and analyzed the data. The main objective in this paper is to clarify the degree of contribution of reading source code for programming learning, to analyze the effectiveness of reading source code, and to evaluate the content of source codes generated by the method in this paper. Therefore, each question did not desire students to ask the effectiveness of the function itself in the developed system such as usability. Namely, students responded to the learning effect itself of source code reading-based learning. As the developed system is only to facilitate the automation of scoring a test, the authors emphasized that each question does not ask the usefulness of the developed system itself. This paper gave 12 questions asking how much reading source code contributed to the programming learning. Student’s responses were obtained with a 6-grade Likert scale. Questions given to students are as follows.

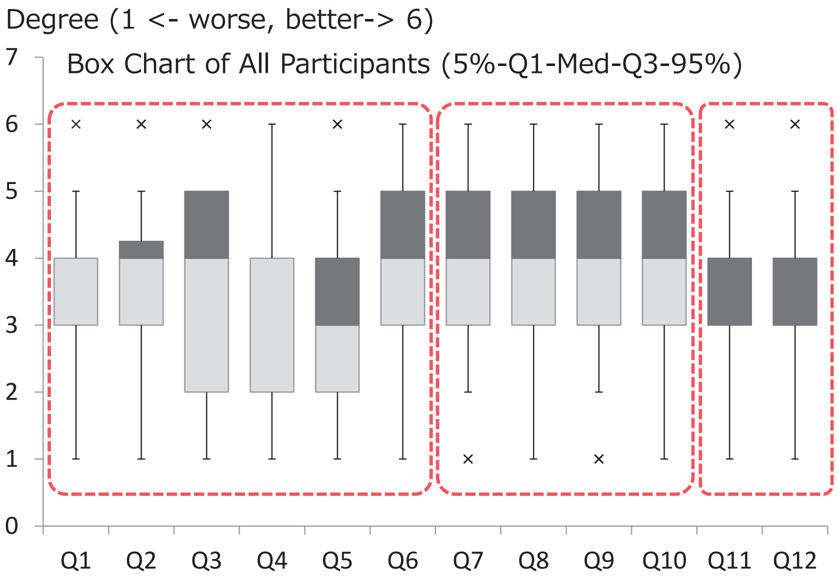


Figure 3: Evaluation result of all students

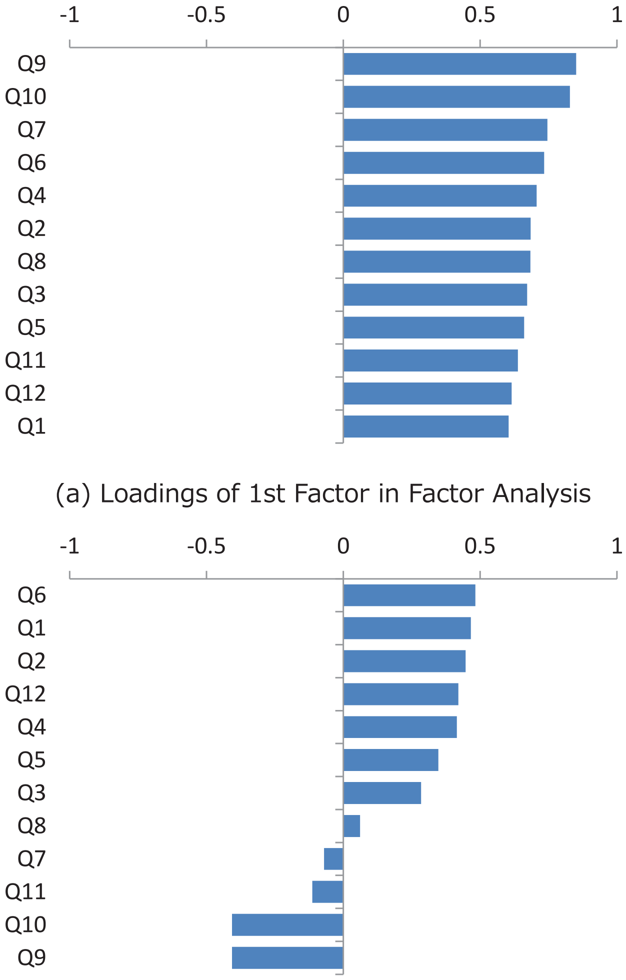
- Q1: The degree of contribution for understanding the fundamental concept of programming.
- Q2: The degree of contribution for understanding the mechanism of assignment operator and the sequence of operation in the assignment operator.
- Q3: The degree of contribution for understanding/giving attention to the advanced knowledge such as compound assignment operator and increment/decrement.
- Q4: The degree of contribution to getting used to programming.
- Q5: The degree of contribution to reading someone's program and fixing a bug.
- Q6: The degree of contribution to getting used to a calculation with remainder operator.
- Q7: Will you be able to read the remainder operator "%" properly if you continue to train reading source code?
- Q8: Will you be able to read the increment/decrement "++, - -" properly if you continue to train reading source code?
- Q9: Will you be able to read the compound assignment operators "+=, -=" properly if you continue to train reading source code?
- Q10: Will you be able to read the compound assignment operators "*=, /=, %=" properly if you continue to train reading source code?
- Q11: Will you be able to deal with the difference in the sequence of operation depending on compilers?
- Q12: The degree of contribution of reading source codes generated by the method in this paper for acquiring knowledge of programming.

Questions Q1-6 are to reveal the contribution of reading source code for programming learning, Q7-10 are to reveal the expectation of reading source code for improving pro-gramming skill, and Q11 and Q12 are to reveal the quality of source code.

The summary of evaluation results is shown in Fig. 3. Since the main objective of this paper is to clarify the analysis result with factor analysis, only the summary of the experimental result is shown below. The details should be found the reference [14][3]. In the result, top 3 scores were Q7 (4.075), Q9 (3.850), and Q8 (3.776), on the other hand, three worst scores were Q5 (3.206), Q12 (3.355), and Q4 (3.411). From the high evaluation results of Q7-Q9, the source code reading-based learning would be strongly expected for improving the reading of remainder operator, increment/decrement and compound assignment operators +=, -=. At the same time, the low results of Q4, Q5 and Q12 suggest that most students would feel that reading automatically generated source codes did not fully contribute to getting used to programming, reading another man's program and fixing a bug. As these reasons, the source codes in this paper were strikingly different from a practical source code and were not the contents for fixing a bug. Reading randomly generated source codes sometimes touched a confusing thing about programming, i.e, the execution result of a complex statement including both a compound assignment operator and increment/decrement differs according to a compiler. For example, the execution result of the source code (f) shown in Fig. 2 depends on the kind of compiler, and also its version. This point would be an extremely troublesome mechanism for students not good at programming. Therefore, the factor of Q12 would give the negative impact for items Q4 and Q5 because source codes including extremely complex statements would cultivate a sense of fear for reading programming.

This paper employed factor analysis for the questionnaire result to examine the consciousness of students for source code reading-based learning. In factor analysis, we gave the initial value of communality by SMC（Squared Multiple Correlation), and removed linearly combined variables. Factors were presumed with the maximum likelihood method and were not rotated. Only the larger factor than one was extracted in the characteristic value of the correlation matrix. Fig. 4 shows the first (a) and the second (b) factor loadings, where the contribution ratio of the first factor was 55.33%, and the second factor was 13.30% respectively. Based on these volumes of contribution ratio, this paper focused on only the first and the second factors. the 1st factor shown in Fig. 4 (a) is probably to be the evaluation for source code reading-based learning with the developed system itself because all factor loadings are positive. It is considered to be natural that students who like/be good at programming got interested in source code reading, so as the result, positive responses would be obtained in whole, and vice versa. Fig. 4 indicates that students who are positive for programming strongly expected the source code reading to equip the concept of compound assignment operators, on the other hand, the students also judged that the source code reading was relatively weak to the factors Q1, Q11, and Q12, which are directly linked with the quality of questions. This may mean that randomly generated source code was not perfect to support source code reading even for positive students to programming.

The second factor is difficult to interpret at first glance. The positive/negative loads in the second factor might not be related to programming skill because they did not depend on the score of programming. As an evidence, Table 2 and Table 3 show the numbers according to the sign of first and second factors. Interestingly, in Fig. 4 (b), the factor loadings about the subjective learning effect (the degree of satisfaction) of the source code reading summarized by Q1-6 were contrary to the factor loadings about the future expectation summarized by Q7-10. Based on this point, the authors assume the second factor as the evaluation for the contents of questions. Students with a positive value in the second factor are considered to be the learners who were satisfied with the source code reading-based learning and felt its effect but whose self-confidence to improve their



(a) Loadings of 1st Factor in Factor Analysis

(b) Loadings of 2nd Factor in Factor Analysis

Figure 4: the first and the second factor loadings

source code reading skill is relatively low. As a reason not to get enough self-confidence, presumably, they were satisfied with the present type of source codes and the amount of learning; on the other hand, they de-sired another kind of question because they already mastered all factors included by the present source codes. Or, this learning helped them to grasp their own skill level (therefore they highly appreciate this learning) but made them realize that their reading skill is still insufficient (therefore they felt uneasy about their own reading skill). To further support them, various kinds of questions are essential to improve their self-confidence. For example, they might need a question to get used to programming, reading someone's program, and fixing a bug. On the contrary, students with the negative value of the second factor are considered to be the learner who wanted to continue to train source code reading skill with the same sort of questions. As the reason why the expectation for learning is high but the contribution degree in current learning is low, we assume that they cannot solve the current problems, but they estimate they will be able to do if continuing. Perhaps they could not be satisfied enough with only the amount of learning given by this paper to overcome their weak points because they might have noticed the depth of programming such as the difference in calculation result

Table 2: The numbers of students according to the values of the first and second factors (only students whose score of achievement test in the programming class is above the average)

	Pos.(first)	Neg.(first)
Pos.(second)	11	11
Neg.(second)	16	13

Table 3: the numbers of students according to the values of the first and second factors (only students whose score of achievement test in the programming class is below the average)

	Pos.(first)	Neg.(first)
Pos.(second)	12	19
Neg.(second)	13	12

depending on a compiler or its version. They felt that the difficulty level of presented some questions was far beyond their ability, so they evaluated the contribution Q1-6 relatively low. Meanwhile, their expectation of source code reading with present type of questions was relatively high. This tendency is assumed to be a firm evidence that they would like to continue learning with the same type of questions in this study and overcome the difficulty point. Based on the consideration, increasing the number of question and providing the variety of learning materials would be an important subject towards the improvement of learner's satisfaction.

Fig. 5 shows the distribution of students' factor scores, where "Upper" denotes the group of students whose score of achievement test in the programming class is above the average (n = 51), and "Lower" denotes other students (n = 56). Only Fig. 5 could not provide enough information to examine the difference of learners depending on the score of the achievement test. Therefore, we give the distribution of each factor score with Tukey's boxplot as shown in Fig. 6. Fig. 6 shows a significant difference in the first factor loading which is statistically checked by Welch's t-test and also we can surely see it. Fig. 6 (b) shows an interesting pattern. Students in the higher group probably required a lot of questions like the present type, which are puzzle-like questions and can provide an opportunity to know sophisticated knowledge, especially compiler dependence behavior. Students in the other group probably required various types of problems including source codes similar to actual source codes, which are useful directly for accumulating experience of actual source code reading such asreading someone's program, and fixing a bug. As mentioned above, by factor analysis, this paper was able to clarify the potential needs depending on student's skill level towards further learning support which would be the significant finding.

5 Conclusion

This paper showed the detail of the source code reading-based learning support system and the analysis result of learner's subjective evaluation result with factor analysis. As the main task, this paper aimed to evaluate the learner's characteristic from the subjective viewpoint on the developed system. The developed system was utilized in a

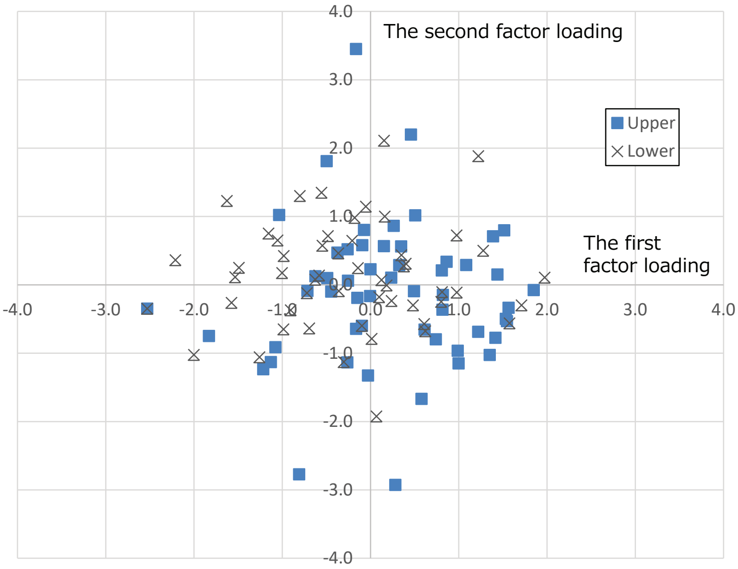


Figure 5: The distribution of students' factor scores

programming course to give the fundamental of C language for freshman students majoring in Informatics, most of them were a beginner of programming. Mini-tests were given several times in the programming lecture by using the capability of automatic generation of source code and obtained each learner's subjective response. This paper obtained 107 student responses from the questionnaire, after the students had completed one semester of the instruction, and an-alyzed the data. From the basis analysis result, it turned out that the reading source code reading-based programming learning would be strongly expected for improving the reading skill of remainder operator, increment/ decrement and compound assignment operators +=, -=. On the other hand, it also turned out that many students felt the automatically generated source codes did not fully contribute to getting used to programming. After the basic analysis, factor analysis was used to analyze the questionnaire result. The analysis result suggested that the first factor is the evaluation for the developed system and the second factor is the evaluation for the contents of questions. Based on the analysis result, this paper concluded that a large amount of questions with various kinds are essential to improve the evaluation of the developed system. By factor analysis, this paper was also able to clarify the potential needs depending on student's skill level towards further learning support which would be the significant finding.

The authors will analyze the learning log data as a future work. The developed system will be certainly efficient to check the understanding level of the programming language specification, the operation rule, the basic memory ability, and the basic calculation ability. Additionally, the developed system would also be useful as a method to grasp what and how much skill is insufficient for improving programming ability because it can collect each learner's response according to the characteristic of a question from daily education. Therefore, a method to judge each student's understanding level of programming and the difficulty of source code will be examined.

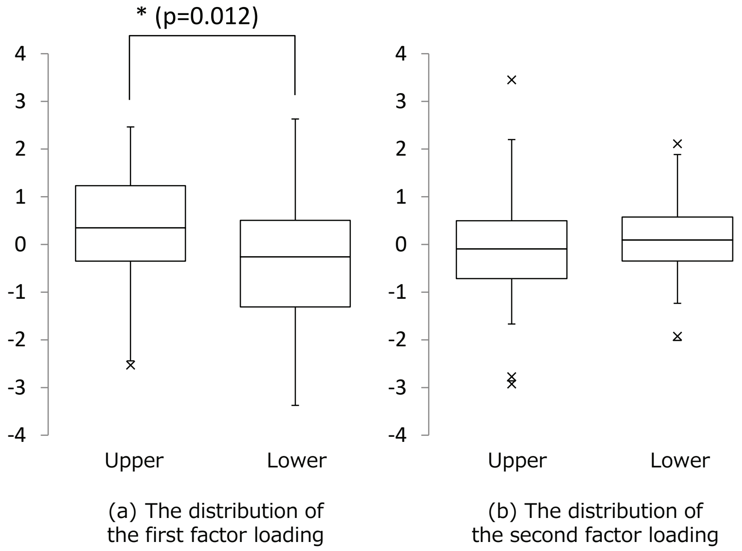


Figure 6: The distribution of each factor score (Tukey's boxplot)

Acknowledgments

This work was partly supported by Japan Society for the Promotion of Science, KAKENHI Grant-in-Aid for Scientific Research(C), No.16K01147 and No.17K01164.

References

[1] A. Robins, J. Rountree and N. Rountree, Learning and teaching programming: a review and discussion. Computer Science Education, Vol.13, No.2, pp.137-172 (2003).

[2] A. Pears, S. Seidman, L. Malmi, L. Mannila, E. Adams, J. Bennedsen, et al., A survey of literature on the teaching of introductory programming, ACM SIGCSE Bulletin, Vol.39, No.4, pp.204-223 (2007).

[3] M. Okamoto, K. Terakawa, M. Murakami, K. Ikeda, M. Mori, T. Uehara and H. Kita, Computer Programming Course Materials for Self-Learning Novices, Proc. of World Conference on Educational Multimedia, Hypermedia and Telecommunications, pp.2855-2861 (2010).

[4] M. Lopez, K. Sutton and T. Clear, Surely we must learn to read before we learn to write!, Proc. of the Eleventh Australasian Conference on Computing Education, Vol.95, pp.165-170 (2009).

[5] H. Kanamori, T. Tomoto and T. Akakura, Development of a Computer Programming Learning Support System Based on Reading Computer Program. In: S. Yamamot (ed.) HCI 2013, Part III. LNCS, Vol.8018, pp.63-69. Springer, Heidelberg (2013)

[6] T. Arai, H. Kanamori, T. Tomoto, Y. Kometani and T. Akakura, Development of a learning support system for source code reading comprehension, In: S. Yamamoto (ed.) HCI 2014, Part II. LNCS, Vol.8522, pp.12-19. Springer, Heidelberg (2014)

[7] M. Tang, Caesar, A social code review tool for programming education, Doctoral dissertation, Massachusetts Institute of Technology (2011).

[8] T. Busjahn and C. Schulte, The use of code reading in teaching programming, Proc. of the 13th Koli Calling International Conference on Computing Education Research, pp.3-11 (2013).

[9] R. Lister, E. Adams , S. Fitzgerald, W. Fone, et al., A multi-national study of reading and tracing skills in novice programmers, ACM SIGCSE Bulletin, Vol.36, No.4,pp.119-150 (2004), DOI: http://dx.doi.org/10.1145/1041624.1041673

[10] K. Cunningham, S. Blanchard, B. Ericson, and M. Guzdial, Using Tracing and Sketching to Solve Programming Problems: Replicating and Extending an Analysis of What Students Draw, In Proceedings of the 2017 ACM Conference on Interna-tional Computing Education Research (ICER ’17), ACM, pp.164-172 (2007), DOI: https://doi.org/10.1145/3105726.3106190

[11] D. Spinellis, Reading, writing and code, ACM Queue, Vol.1, No.7, pp.84-89 (2003).

[12] M. Weiser, Programmers Use Slices When Debugging, Communications of the ACM, Vol.25, No.7, pp.446-452 (1982).

[13] D. Boswell, T. Foucher, The Art of Readable Code (Theory in Practice), O’Reilly Media (2011).

[14] S. Matsumoto, K. Okimoto, T. Kashima and S. Yamagishi, Automatic Generation of C Source Code for Novice Programming Education, Human-Computer Interaction. Theory, Design, Development and Practice: 18th International Conference, HCI Inter-national 2016, Toronto, ON, Canada, July 17-22, 2016. Proceedings, Part I, Springer (2016).