Understanding Level Analysis of Students using Programming Education Support Tool pgtracer

Miyuki Murata
Faculty of Liberal Studies
National Institute of Technology, Kumamoto College
Yatsushiro, Japan
m-murata@kumamoto-nct.ac.jp

Tetsuro Kakeshita
Graduate School of Information Science
Saga University
Saga, Japan
kake@is.saga-u.ac.jp

***Abstract*—We are developing a programming education support tool pgtracer. Pgtracer provides fill-in-the-blank questions to the students and collects student log to analyze student's learning process and understanding level. In this paper, we utilize pgtracer at an actual programming course and analyze the collected log. There is a negative correlation between required time and right answer ratio of a blank. Although there is a correlation between the mid-term examination score and pgtracer score, pgtracer score is useful to distinguish programming ability of the students. We also perform survey and interview to the students about the usefulness of pgtracer and fill-in-the-blank questions.**

Keywords— Learning Analytics (LA); computer programing education; e-learning; Moodle; fill-in-the-blank question

I. INTRODUCTION

Computer programming is essential at national institute of technology and university majored in science and technology in order to fully make use of a computer and information technology. However we often find students with low programming skill and/or low motivation to learn computer programming at an actual class. Individual guidance is desirable for such students. However there is a limitation of the teaching staff and budget.

Considering the above situation, we are developing a programming education support tool named pgtracer [1]. Pgtracer provides fill-in-the-blank questions for programs and corresponding trace tables representing the execution order of program steps and the value of the variables at each step. The teacher can define various types of blanks within the program and trace table so that difficulty level of the problem can be easily adjusted. Pgtracer automatically evaluate the filled program and trace table when a student fills the blanks and submit the answer. Thus a student can learn computer programming at any time and place as long as the internet connection and PC are provided.

Pgtracer also collects student log at each time a student fills a blank. In addition, pgtracer provides variable student log analysis functions [2]. Using these functions, a teacher can quantitatively recognize students achievement level.

In this paper, we utilize pgtracer at an actual programming course at Kumamoto National Institute of Technology. The research questions are to find tendency of student's answering behavior from the log collected by pgtracer and to detect topics that student understanding is not enough.

Fill-in-the-blank questions are utilized as homework assignment. We analyze the collected log to recognize programming ability and learning motivation of each student. We also interviewed some of the students in order to validate our recognition.

Funabiki et. al. proposed a blank element selection algorithm and developed JPLAS to provide fill-in-the-blank questions of Java programs [3]. However JPLAS does not support trace table and does not provide log analysis functions. Currently we are developing pgtracer for Java programs.

Itado et al. proposed a method to evaluate learner's ability from exercise sentence sorting or corresponding coding [4]. This method uses exercise scores. But it does not consider required time and answering process. Malliarakis et al. proposed a framework that guides incorporation of learning analytics mechanisms in computer programming education [5]. However the detail of the collected data is not presented. Helminen et al. developed a web-based Python programming environment which collects and analyzes learner's programming process [6]. This environment also does not provide a function to estimate student's achievement level.

This paper is organized as follows. The next section introduce major functions of pgtracer. We then explain the experiment plan in Section III. In Section IV, we analyze and discuss the required time and right answer ratio of each question. Correlation between pgtracer score and mid-term examination score is analyzed in Section V. We analyze the difficulty level of each blank in Section VI. Result of the student interview is reported and discussed in Section VII.

II. PROGRAMMING EDUCATION SUPPORT TOOL PGTRACER

Programming education support tool pgtracer is developed as a plug-in of a well-known learning management system Moodle.

Pgtracer provides various functions to the students and teachers. A teacher can create and edit fill-in-the-blank questions using pgtracer. When a student selects a question and fills the blanks, pgtracer automatically executes the filled

This research is supported by JSPS KAKENHI under grant number 16K01022 and 17K01036.

program and compares the result with the filled trace table. The student answer is evaluated based on the comparison result.

Pgtracer automatically collects the student log each time the student fills a blank. The collected log is stored in a MySQL table. A log record contains student id, question number, place of blank, date, time, student answer and the evaluation result. Pgtracer provides various types of the data analysis functions to analyze the collected log.

The student utilization function (Fig. 1) illustrates the overall behavior of the student in terms of the average score and required time to answer the questions. This function is useful to understand the overall achievement and workload of the students.

The analysis function of a student (Fig. 2) summarizes the achievement of the selected student. Note that the actual student name is excluded from the figure because of the privacy reason. This function is useful to understand the achievement of each student.

The analysis function of a question (Fig. 3) illustrates the averages and distributions of the student score and required time. The averages and distributions are separately calculated for the case of the first attempt and the case of the maximum score. The average and distribution for the first attempt correspond to the actual ability of the students, while those for the maximum score can be used to recognize students' effort since many of the students repeat the trial until they earned the 100% score.

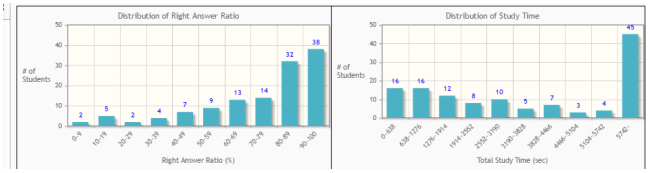

Fig. 1 Student Utilization

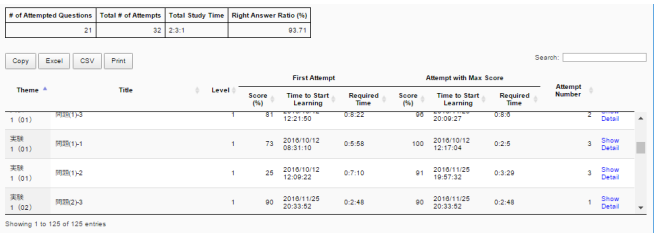

Fig. 2 Analysis Function of a Student

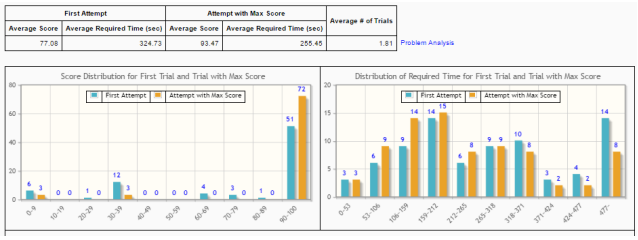

Fig. 3 Analysis Function of a Question

The analysis function of a blank within a question (Fig. 4) represents the right answer ratio and average required time for each blank of a question. Then it is possible to recognize difficulty level of the blanks. Each of the blanks is assigned a unique number as illustrated.

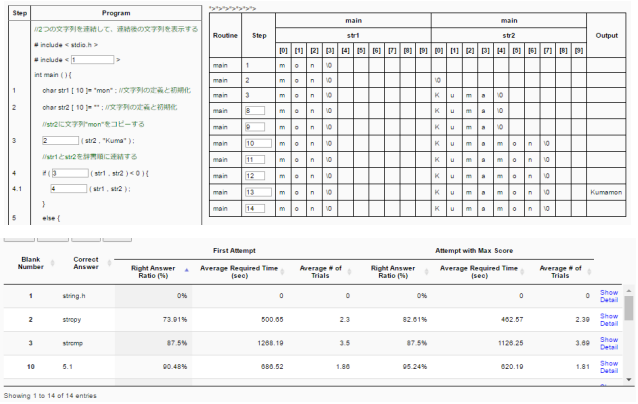

Fig. 4 Analysis Function of a Blank within a Question

The analysis function of an answer process (Fig. 5) represents the intermediate state of a selected student answer. The upper table represents the sequence of filled blanks of the selected answer associated with the required time to fill each blank. When a teacher selects a blank, pgtracer illustrates the state of the student answer when the student filled the selected blank at the lower table representing the question. The yellow blank is the one recently updated.

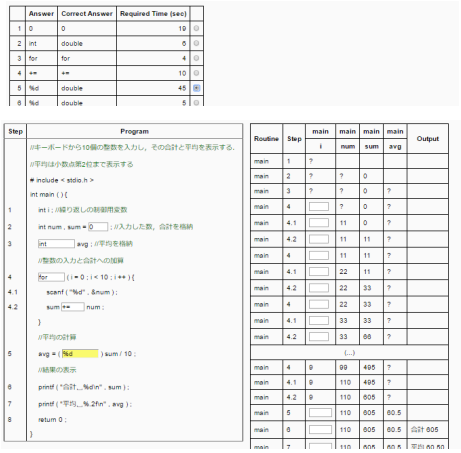

Fig. 5 Analysis Function of an Answer Process

These analysis functions are fully utilized to understand the student achievement and behavior of the experiment described in this paper. Although these analysis functions are provided for the teachers, some of the functions are also provided to the students for self-analysis of a student.

III. Experiment Plan

The experiment was performed for the 218 students at the second academic year of Kumamoto national institute of technology from October 2016 to February 2017. The students are majored in one of the following areas: mechanical engineering, electronics, civil engineering, architecture, bio technology and chemistry. Although they are not majored in IT, the students are learning computer programming using C language at the common course named "Fundamental of

TABLE I
AVERAGE SCORES AND REQUIRED TIME OF THE QUESTIONS

Question	Description	First Attempt		Number of Students
		Average Score (%)	Required Time (sec)	
(1)-1*	Calculate and print sum, difference, multiplication, division and reminder of two integers	77.08	324.73	78
(1)-2*	Show absolute value of the input integer and whether it is even or odd.	67.20	438.10	73
(1)-3*	Calculate and print 1-th power to 5-th power to the input integer.	56.13	627.13	69
(2)-1	Calculate and print sum and mean value of the 10 input integers.	58.13	412.66	64
(2)-2	Store 10 integers to an array and print them in the reverse order.	69.88	306.24	58
(2)-3	Show the string stored in a character array by converting upper case letter to lower case and vice versa	56.31	514.64	56
(3)-1	Show the number of negative integers until 5 positive integers are found.	63.25	332.85	47
(3)-2	Concatenate and print two strings	43.75	349.42	43
(3)-3	Show the point having the largest distance from the origin among three.	52.14	361.74	42
(4)-1	Calculate and store 2 to the 0-th power to the 8-th power and show the selected value.	57.25	395.65	40
(4)-2	Show the length of the strings until "end" is detected.	52.55	410.71	38
(4)-3	Calculate and print sum of the edge length of four triangles stored in a two-dimensional array.	70.41	538.64	33
(5)-1	Same as (3)-3 but we delete comments explaining variables	58.84	193.81	36
(5)-2	Assignment and print variables through pointer.	79.81	181.44	36
(5)-3*	Add two float variables using pointer.	75.65	136.78	36
(6)-1	Same as (4)-3 but we delete comments explaining variables	68.95	347.81	21
(6)-2	Calculate and show area of a circle from the diameter using pointer.	67.45	335.21	19
(6)-3	Print array elements using pointer.	52.11	552.61	18

* contain log data of a test user.

Computer Science". The course is composed of 30 weeks and 90 minutes class each week. The course is provided by two teachers. One is giving lecture and exercise. The other, one of the author of this paper, is supporting the course by utilizing pgtracer for the department of biological and chemical systems engineering.

Students have learned variables, standard I/O, conditional and iterative statements in the previous semester. At the current semester, students learn array, pointer and function during the experiment.

We introduced pgtracer to the student at the end of June 2016 and registered the students to Moodle at the beginning of July 2016. We provided 18 questions, listed in Table 1, and announced the students to learn computer programming by utilizing pgtracer.

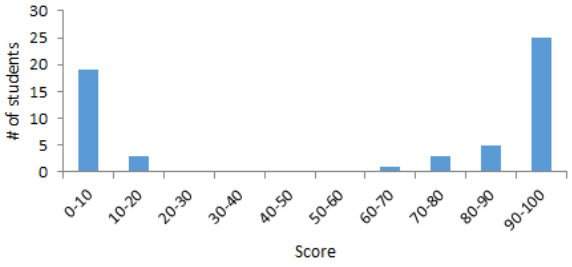

Fig. 6 Score Distribution for the First Attempt of Question (2)-3

IV. ANALYSIS OF REQUIRED TIME AND RIGHT ANSWER RATIO

Table 1 represents scores and required times for the 18 questions for the first attempt. Fig. 6 represents the score distribution of question (2)-3. The readers can observe twin peaks of the distribution at 0-10 and 90-100. Similar distribution was observed at many of the questions. The peak at 90-100 implies that the question itself was not difficult. The peak at 0-10 means that some students did not actually solve it.

We find two patterns of the required time distribution. One is the distribution pattern with twin peaks at the shortest and the longest time intervals (Fig. 7). The other is the pattern with a single peak at the middle of the time interval (Fig. 8).

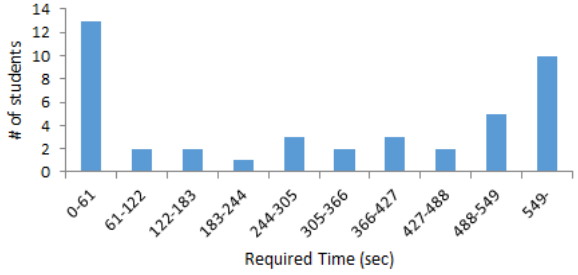

Fig. 7 Distribution Pattern with Twin Peaks of the Required Time

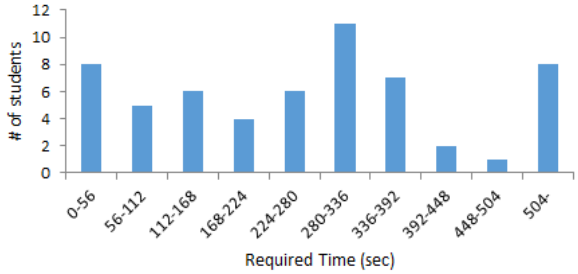

Fig. 8 Distribution Pattern with a Single Peak of the Required Time

At first let us discuss the pattern with twin peaks. There are four questions with the distribution pattern of this type. Table 2 represents such questions and the number of empty answers

TABLE 3
THE BLANKS WHICH THE RIGHT ANSWER RATIO IS LESS THAN 60%

Question (1)-3	Right Answer Ratio	43.2%
	Place of Blank	ans = ans * n within a for statement
	Typical Wrong Answers	No Answer (13 answers), n^i (4 answers), *i, *n, n*, etc.
Question (2)-3	Right Answer Ratio	53.3%
	Place of Blank	s[i]='\0' within a while statement
	Typical Wrong Answers	No Answer (8 answers), \0, 10, EOF, etc.
Question (4)-1	Right Answer Ratio	41.7%
	Place of Blank	Column to show output result within trace table
	Typical Wrong Answers	No Answer (6 answers), 024, 16, 64, 256, etc.

within the answers which required less than two minutes. The number of empty answers are more than 58% for these questions. This means that the students just checked the question without solving it. The readers can observe that the average score of these questions are relatively low among the 18 questions listed in Table 2. Also considering the distribution observed at Fig. 6, we guess that this is because the teacher announced that some of the questions are used at the mid-term examination.

TABLE 2
NUMBER OF EMPTY ANSWERS WITHIN THE ANSWERS WHICH REQUIRED LESS THAN 2 MINUTES FOR THE QUESTIONS

Question	# of Answers require less than 2 minutes	# of Empty Answers
(2)－3	12	7 (58.3%)
(3)－1	12	9 (75.0%)
(3)－2	15	13 (86.7%)
(3)－3	16	13 (81.3%)

The average score of the above students at the mid-term examination is 71.6%. It is lower than the average score (73.3%) of the all students. This means that majority of such students have low programming skill so that they cannot solve the actual problem at the mid-term examination even if they observed the questions using pgtracer in advance.

Next we shall discuss the answers which require more than 10 minutes. There is an answer for question (2)-1 which require 1 hour 42 minutes 24 seconds. However observing the corresponding answer process, we find that the student did not fill any blank after 2 minutes 26 seconds. Thus we can conclude that the student stopped learning. Similar type of answer processes are observed in questions (3)-1, (3)-2 and (3)-3. These answers are excluded from the succeeding analysis.

We shall next discuss the pattern with a single peak. There are three questions of this type: questions (2)-2, (5)-2 and (5)-3. The ratio of the answers whose score is less than 10% is at most 20% in each case. This implies that the students can correctly solve the questions as long as they spend reasonable time. Question (2)-2 is a typical one which the program can be found in the textbook as an example. Similarly (5)-2 and (5)-3 are elementary questions on pointers. As the readers can find from Table 1, the average scores of these questions are more than 70% and are relatively high among the 18 questions.

On the other hand, average scores of the four questions listed in Table 2 are relatively low among the 18 questions in Table 1. These questions are rather difficult from the viewpoint of the students. In such a case, some of the students tend to give up answering the questions. This leads to the twin peaks founded in the time and score distributions.

V. CORRELATION ANALYSIS BETWEEN MID-TERM EXAMINATION AND PGTRACER EXERCISE SCORE

We shall analyze the relationship between the mid-term examination and the exercise using pgtracer in this section.

Fig. 9 represents the relationship between the mid-term examination and the pgtracer scored. The correlation coefficient between these two values is 0.39, which means that there is a weak correlation between them.

Mid-term examination score is high but the pgtracer score is low for the two students within the red circle. Average required time to solve a question is 58 seconds for one student, which is far less than the average required time. The other student answers to the 5 questions within 2 minutes among the 11 questions he solved. These facts indicate that the learning attitude of these two students are not positive.

On the other hand, the average required time is more than 6 minutes for the two students within the green circle. These two students rewrite the same blanks many times while solving the pgtracer exercise. The fact indicates that the student is making trial and error while solving the questions. We can say that the programming skill of these students is not high because a skilled students do not require trial and error. However we can say that the learning attitude of these students are quite positive.

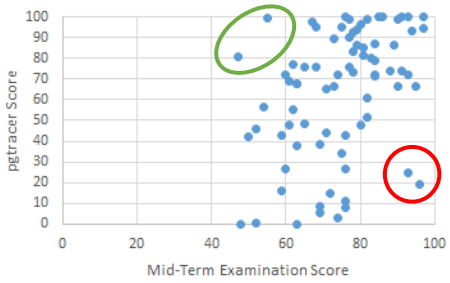

Fig. 9 Relationship between Mid-Term Examination Score and pgtracer Score

Fig. 10 represents the relationship between the mid-term examination score and the total learning time using pgtracer. The correlation coefficient is -0.03, which means almost no correlation. It is expected that there is a strong correlation between the effective learning time and examination score. In the case that a student leaves pgtracer after starting to solve a fill-in-the-blank question, however, the waiting time of pgtracer is included in the learning time so that the learning

TABLE 4
ACHIEVEMENT SUMMARY OF THE STUDENTS SELECTED FOR THE INTERVIEW

Selection Criteria	Student Number	Mid-Term Exam. Score	Average pgtracer Score	Average Required Time to Solve a Question	Total Number of Tried Questions	Total Number of Trials
1	1	96	19.0	58.3 sec.	15	39
1	2	93	24.7	242.2 sec.	15	17
2,3	3	55	99.3	370.0 sec.	20	29
2	5	68	94.8	716.5 sec.	6	15
2	6	67	97.9	366.1 sec.	18	27
3	7	61	69.0	1350.5 sec.	30	82
3	8	82	98.7	398.1 sec.	23	36
3	9	76	42.5	230.7 sec.	15	74

time may not represent the effective learning time. The authors think that it is necessary to analyze the learning process of the students in order to estimate the effective learning time of the students.

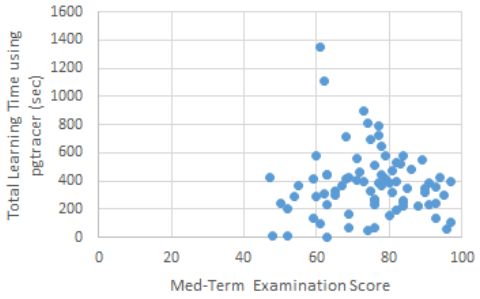

Fig. 10 Relationship between Mid-Term Examination Score and Total Learning Time using pgtracer

VI. DIFFICULTY ANALYSIS OF BLANKS

In this section, we shall analyze the difficulty level of each blank. Basically the difficulty level of a blank can be estimated using the right answer ratio or the required time to fill the blank.

We shall first analyze 4 questions listed in Table 2 whose average required time is more than 10 minutes.

The blanks defined in a do-while statement require the longest time in question (3)-1. Frequency to use a do-while statement is not high so that understanding level of the students tend to be low. We defined two blanks for "do" and "while" within the program. 20% of the students' answers are incorrect. Typical mistakes are confusion with for or while statements.

The blanks to ask correct names of the library functions related to <string.h> require the longest time in question (3)-2. Typical mistakes are the misspelling of the function names. Actually the average score of question (3)-2 is the lowest among the 18 questions in Table 1.

The blanks to ask correct indices of the array require the longest time in question (3)-3. The right answer ratios are around 85%. Students need to understand the position of each element within the array in order to correctly fill the blanks.

We shall next analyze the blanks whose right answer ratio is less than 60% illustrated in Table 3. The right answer ratio is defined by the number of right answers divided by the total number of answers filled to a blank. The "blank" answers are excluded from the division.

The extracted blank in question (1)-3 is a typical one using the for statement. Typical wrong answers are the incorrect use of the power operator by confusing the operator defined in Microsoft Excel. Students are also required to understand the iterative algorithm to calculate the power of the input value. Such combination of algorithm and operator is a typical place where inexperienced students are confused.

The blanks to ask the termination condition of a string require the longest time in question (2)-3. 46.7% of the answers are incorrect. Students could not write anything for 50% of such incorrect answers. Although question (2)-3 is a typical one for string manipulation using while statement, we found the students who did not precisely memorize termination symbol of a string literal.

The blanks to ask an output value within the trace table require the longest time in question (4)-1. Students are required to understand trace table to correctly fill the blank. However we find some students who do not understand the trace table. The details are explained in the next section.

The explained topics that the achievement of the students is low can be regarded as candidates of future educational improvement. Pgtracer is thus useful to quantitatively analyze student achievement as explained in this section.

VII. SURVEY AND INTERVIEW TO THE STUDENTS

We selected 9 students based on the following criteria and conducted the individual interview to the students after the experiment. The purpose of the interview is to validate the analysis result using pgtracer.

1. Students who obtained a high score at the mid-term examination but the pgtracer score is low. (Criteria 1)
2. Students who obtained a high pgtracer score but the mid-term examination score is low. (Criteria 2)
3. Students frequently using pgtracer. (Criteria 3)

Table 4 represents the achievement summary of the students. The red numbers represents the selection criteria of each student. The total number of trials include the number of trials to the tutorial questions as well as the 18 problems for the exercise.

All of the students said that pgtracer problems are suitable to learn computer programming compared with traditional programming assignment which create computer program from the scratch. Pgtracer can automatically evaluate each blank and show the evaluation result just after a student fills the

blank. This function is particularly effective to increase motivation to learning. Students confirm basic grammars of the program and understand behavior of the provided program using pgtracer.

Two of the students selected based on criteria 2 solve pgtracer questions with their friends. Thus there is a case that pgtracer score do not properly represents programming skill of the student.

On the other hand, the students selected based on criteria 1 did not fill all the blanks of the tried questions. One use pgtracer only to view the problems before the mid-term examination. The other had a difficulty in understanding trace table.

Many of the student initially did not understand the concept of trace table. There is also a case that the entire columns of the trace table could not be shown in the computer screen.

There is also a student No. 7 who quickly understand trace table and fills the blanks within the program by utilizing the information shown in the trace table. He obtained a high score (82%) at the final evaluation. Considering our experience of the past experiment, there is a correlation between the understanding level of the trace table and the programming skill.

We find some answer processes that a student repeatedly fills the same blank with different values as illustrated in Fig. 11. The strings within a red rectangle represent a sequence of student answers to a blank. The strings within the blue rectangle represent a sequence of student answers to a consecutive blanks of the same row within the trace table.

Such blanks often coincide with the blanks which the student feels his/her insufficient understanding of the program. In such cases, students tend to use trial and error strategy to get a better score. Thus there is a possibility to detect the corresponding blanks by analyzing the student log.

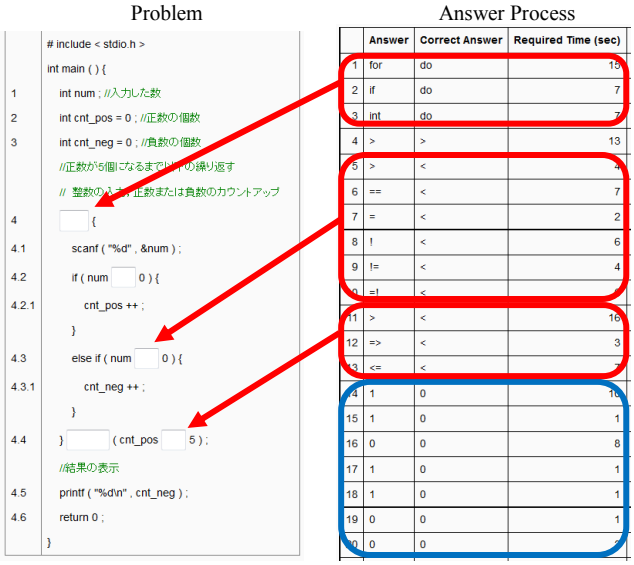

Fig. 11 Repetitive Answer Process for Question (3)-1

VIII. Conclusion and Future Work

In this paper, we analyze the student behavior to solve fill-in-the-blank programming questions using pgtracer.

We detect various topics that the understanding of the students is not enough by analyzing the blanks with a low right answer ratio or a long required time compared with those of the other blanks. We also find the tendency that the students repeatedly fills the same blank until they obtain a satisfactory score even when their understanding level is not enough.

The strings filled to such blanks represents the range of students' retrieval so that we can expect to have various information about the topics where the students do not understand well or the topics where the students understand incorrectly by analyzing the candidate answers they used. The information will be useful for a teacher of programming class to improve programming education. And also students can improve their programming skill by practicing the weak points that are detected in the information.

Currently the analysis functions of pgtracer utilize final student answers to each of the blanks. We are planning to analyze intermediate student answers in order to improve pgtracer. This improvement will be useful in order that pgtracer provides more valuable information to understand student programming skills.

Acknowledgment

We are grateful to the students of the Kumamoto National College of Technology for their cooperation to our experiment. We also appreciate Professor Tetsuya Yonezawa for his cooperation of the Fundamental of Computer Science to our experiment.

References

[1] T. Kakeshita, R. Yanagita, K. Ohta, "Development and evaluation of programming education support tool pgtracer utilizing fill-in-the-blank question", *Journal of Information Processing: Computer and Education*, Vol. 2, No. 2, pp. 20-36, Oct. 2016. (in Japanese)

[2] T. Kakeshita, K. Ohta, "Student log analysis functions for web-based programming education support tool pgtracer", 17th International Conference on Information Integration and Web-based Applications & Services (iiWAS2015), Brussels, Bergium, pp. 120-128, Dec. 2015.

[3] N. Funabiki, et al., "Analysis of fill-in-the-blank problem solutions and extensions of blank elent selection algorithm for Java programming learning assistant system", World Congress on Engineering and Computer Science (WCECS 2016), San Francisco, USA, pp. 237-242, Oct. 2016.

[4] Y. Itado, F. Harada and H. Simakawa, "Judgement of Learner Ability from Exercise Sentence Sorting and Corresponding Coding", Forum on Information Technology (FIT 2013) K-035, pp. 635–638, 2013. (in Japanese)

[5] C. Malliarakis, M. Satratzemi and S. Xinogalos, "Intergrating learning analytics in an educational MMORPG for computer programming", IEEE 14th Int. Conf. of Advanced Learning Technologies, pp. 233–237, 2014.

[6] J. Helminen, P. Ihantola and V. Karavirta, "Recording and analyzing in-browser programming sessions", 13th Koli Calling Int. Conf. on Computing Education Research, pp. 13-22, 2013.