An Online Automated Scoring System for Java Programming Assignments

Hiroki Kitaya and Ushio Inoue

Abstract—**This paper proposes a web-based system that automatically scores programming assignments for students. The system receives a Java application program submitted by a student and returns the test result immediately. The test consists of compiler check, JUnit test, and result test. The result test is very useful for students in elementary programming courses, because a typical program is composed of only a main method that reads/writes data from/to the standard input/output devices. The system is used experimentally in an actual course of our university. We confirmed that the accuracy of scoring is very high and the response time of the system is satisfactory.**

Index Terms—**Automated scoring, java programming assignments, web systems, learning systems.**

I. Introduction

Many students learn a programming language at introductory classes of universities. In such a class, students are imposed programming assignments by the lecturer to improve their skills. The lecturer usually checks the student programs manually and returns feedbacks to the students. It becomes a heavy burden for the lecturer if there are many students in the class. The lecturer may also make mistakes in the process. On the other hand, each student in the class cannot know if his/her program is correct or not until the feedback is returned. Some of the students may even forget the detail of his/her program. This reduces the efficiency of improvement of programming skills for many students.

In this paper, we propose a web-based scoring system, which accepts submissions of programming assignments, tests automatically the submitted programs, and returns feedbacks to the students immediately. The system is targeted at Java language of an introductory level. A typical program at this level is composed of a single class with only a main method, which reads data from the standard input and writes results to the standard output. Traditional unit test methods such as JUnit cannot test such a program because there is no method that takes arguments and returns values. To solve the problem, our system uses the Regular expression and Reflection APIs of Java.

The rest of the paper is organized as follows: Section II describes related work. Section III defines the target and goal of the research. Section IV proposes our method. Section V gives the implementation of our method in detail. Section VI describes the evaluation results in our actual class. Section VII concludes the discussion.

II. Related Work

Various research papers have been published on automated programming scoring systems.

Christoph *et al*. [1] presented an approach to the automatic revision of homework assignments in programming language courses, particularly exercises in Prolog and Scheme. They provided an abstract frame in the AT(x) (analyze-and-test for a language x) system, which analyzes programs written by a student, and sends comments back to the student via Web Assign [2]. J. L. F. Alemán [3] presented an experience using automatic assessment in a programming tools course. He extended the traditional use of an online judging system with a series of assignments related to programming tools. His research suggested that automated assessment systems promoted the students interest and produced statistically significant differences in the scores between experimental and control groups. Singh *et al*. [4] proposed a method for automatically providing feedback for introductory Python programming problems. Their system derives minimal corrections to student’s incorrect solutions based on a reference implementation of the assignment and an error model that students might make. Jinrong *et al*. [5] designed and implemented the semantic matching based automatic scoring system for C programming language. Their system standardizes student programs and template programs, and then calculates their semantic similarity, so as to score student programs. By their system, instructors were freed from boring and tedious marking work, and scores of the students became more fair and reasonable. Shamsi and Elnagar [6] presented a grading system for Java introductory programming courses. Their system grades submissions both dynamically based on the JUnit framework and statically based on the graph representation of the program. The target of their system is similar to our system, but it is rather complex.

We adopted a simpler approach based on the comparison of output text of a reference program and student programs. This is because our target programs are very simple and it is important to indicate an error as quickly as possible rather than analyzing the cause of the error in detail.

III. Prerequisite

A. The Target of This Research

We have developed our method for an introductory Java language course of Tokyo Denki University. Approximately 150 students learn Java in the course for 4 months and most of

Manuscript received September 10, 2014; revised November 11, 2014.

The authors are with Graduate School of Engineering, Tokyo Denki University, Tokyo, Japan (e-mail: 13kmc09@ms.dendai.ac.jp, inoue@c.dendai.ac.jp).

DOI: 10.7763/IJIET.2016.V6.699

the students have less than one year experience of programming. Therefore, homework assigned to the students is writing a simple Java application that input data from the keyboard and display results of computation on the screen.

Table I shows examples of homework actually assigned in 2014.

TABLE I: EXAMPLES OF ACTUAL ASSIGNMENTS

ID	Question
1	Write a main method that reads 5 integers and prints the average excluding the max and min values.
2	Write a main method that reads 3 positive integers and prints the GCD and LCM numbers.
3	Write a method that receives 2 arrays of integer as its arguments and returns common elements in a new array.
4	Write a class that represents the duration of time with specified fields, constructors and methods.
5	Write a tiny application of a single-player poker-game using a given Card class.
6	Write a very simple priority queue class extending a given Queue class.

B. *The Requirements to Our System*

In order to test such student programs as shown in Table I, our scoring system should support the following features:

- Console based test, which compares output data of a student program and the reference program for several test cases.
- Function based test, which calls constructors and methods in a target class, and checks if the returned values of the methods satisfy assertion statements.
- Source code based test, which checks if the code is syntactically and grammatically correct, and if specified classes or methods are properly used in the code.
- Uploading student programs, compiling them with necessary external classes, and displaying the results on the screen in real time.

IV. PROPOSED METHOD

In order to satisfy the above-mentioned requirements, our system tests student programs by the following way shown in Fig. 1.

A. *Compiler Check*

If there are syntactical or grammatical errors in the source code, it is not possible to test the program accurately. Therefore, our system tries to compile a student program first of all. If the compilation fails by any reasons, the error messages of the compiler are displayed to the student after stopping scoring process.

B. *JUnit Test*

If the assignment demands to write a class or a method except for a main, JUnit test is used. JUnit is a framework to write repeatable test methods, each of which contains assertions for testing expected results. It is ordinary used to test public classes and methods, but it also can be used for private classes and methods with the reflection API. Our system executes test methods provided by the instructor.

C. *Result Test*

Every program written for assignments is expected to input data from the keyboard and to display the result on the screen. It is usually done by a main method written by a student or given by the instructor. We can conduct a test by comparing the results of a student program and the reference program provided by the instructor. Our system uses the redirection to supply the input data and to obtain the result, and then compares the results by means the regular expression API.

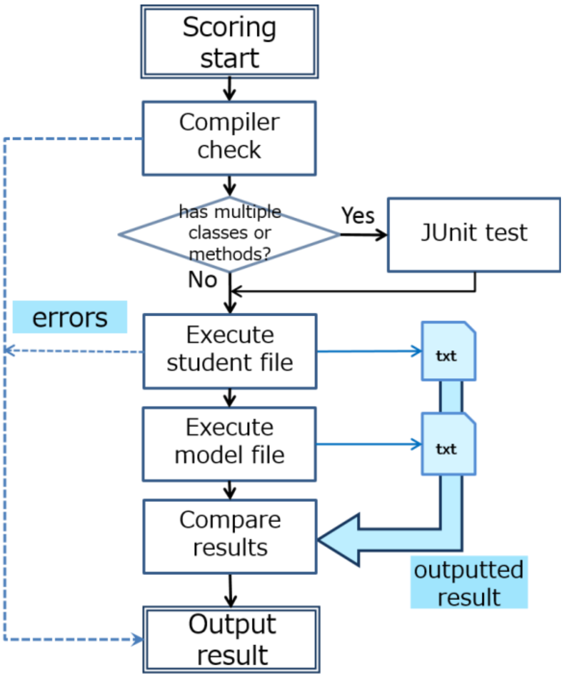


Fig. 1. Process flow for scoring.

V. DESCRIPTION OF THE SYSTEM

A. *Platform and Framework*

We constructed our system on a web site so that students can submit their programs anywhere and get feedback immediately. Our system uses JSP and Java Servlet on Apache Tomcat, which is a servlet container developed by the Apache Software Foundation. In addition, the system uses Struts 2, which is an MVC (Model-View-Controller) framework, developed by the Apache Software Foundation also. Moreover, the system uses MySQL in order to store information of students and assignments.

B. *System Organization*

The structure of our system is shown in Fig. 2. By using Struts 2, the system is composed of three components; "Model" stores and manages data by MySQL, "View" creates a Web page by JSP, and "Controller" receives and tests a student program, and records the result. The following subsections will explain the process of the system.

C. *Uploading Files*

Students submit their assignments by uploading their java source files. The submission page is shown in Fig. 3.

Struts 2 treats the multi-byte data sent from HTML forms in Action class. By using the same name that is specified in the form tag, the file name and contents are acquirable.

A list of assignments is displayed with the latest timestamp if it was already submitted. The student chooses an assignment to submit from the list, and uploads his/her source file by clicking the upload button. The student may resubmit his/her files any number of times until the deadline of the assignment. The evaluate button next to the upload button is used to show the test result of the submitted program.

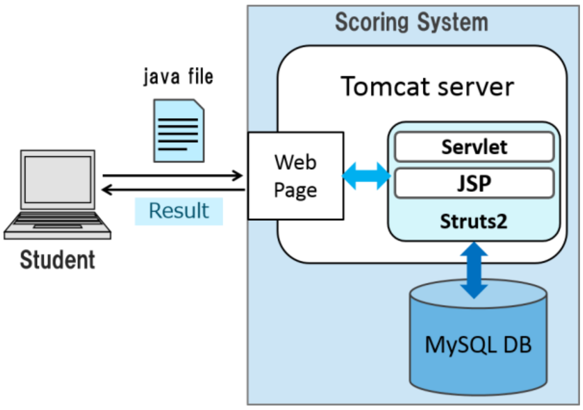
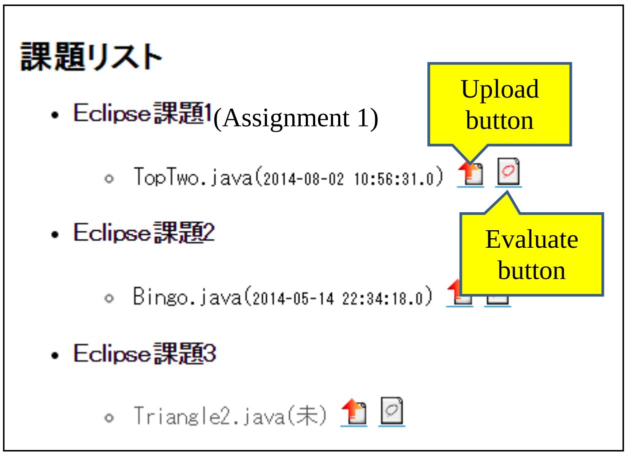


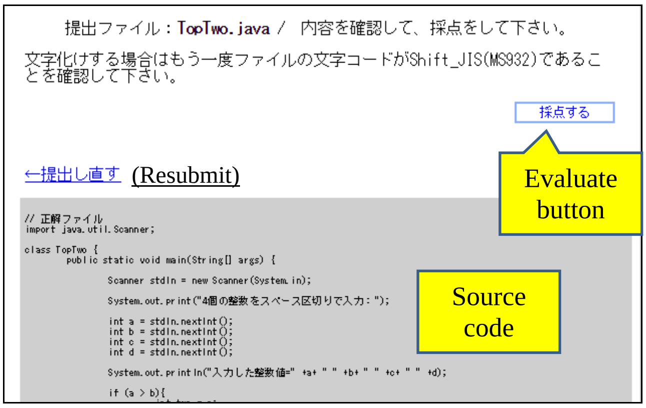
Fig. 2. System structure.



課題リスト

- Eclipse課題1(Assignment 1)
  - TopTwo.java(2014-08-02 10:56:31.0)
- Eclipse課題2
  - Bingo.java(2014-05-14 22:34:18.0)
- Eclipse課題3
  - Triangle2.java(未)

Fig. 3. Submission page.



提出ファイル：TopTwo.java / 内容を確認して、採点をして下さい。

文字化けする場合はもう一度ファイルの文字コードがShift_JIS(MS932)であることを確認して下さい。

```
// 正解ファイル
import java.util.Scanner;

class TopTwo {
        public static void main(String[] args) {

                Scanner stdIn = new Scanner(System.in);

                System.out.print("4個の整数をスペース区切りで入力：");

                int a = stdIn.nextInt();
                int b = stdIn.nextInt();
                int c = stdIn.nextInt();
                int d = stdIn.nextInt();

                System.out.println("入力した整数値=" +a+ " " +b+ " " +c+ " " +d);

                if (a > b){
```

Fig. 4. Confirmation page.

The confirmation page displayed by clicking the upload button is shown in Fig. 4. This page shows the source code in the submitted file so that the student can confirm if the right file has been submitted. The evaluate button displayed on the upper right of the page has the same functionality of that in the submission page.

D. Testing Programs

The Java files uploaded by students are tested by the series of procedure shown previously in Fig. 1. The directory hierarchy for carrying out the compiler check, JUnit test, and result test is shown in Fig. 5.

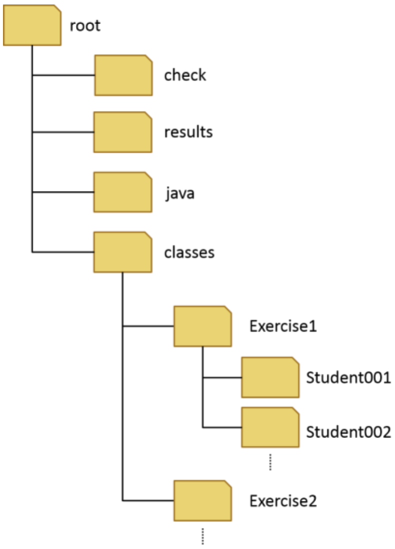


Fig. 5. Directory hierarchy.

Although not described in Fig. 5, the structures inside the “results” and “java” directories are the same as that of “classes” directory. By storing files into separate directories for every assignment and student, the system can test each student file independently without interference of another assignment or student.

1) Compiler check

First, a Java source file uploaded by a student is stored in the “java” directory. Then, the source file is compiled by the Java Compiler API and the class file is saved in the “classes” directory. In the case when the assignment requires external classes that are not contained in the uploaded file, the instructor stores required class files in the “check” directory in advance. If an error occurs while the compiling process, the system saves the diagnostic information on the error at “results” directory.

2) JUnit test

For an assignment that needs writing a new class or method, the system conducts JUnit tests by using the Java Reflection API. The Reflection API treats a class or method dynamically including private resources. The target class is dynamically loaded by the Class Loader and judged whether the program is written according to the specification of the assignment. Therefore, a broad testing can be attained to improve the accuracy of the test results.

If an error is found at this process, the system records the error information, but it goes to the next step.

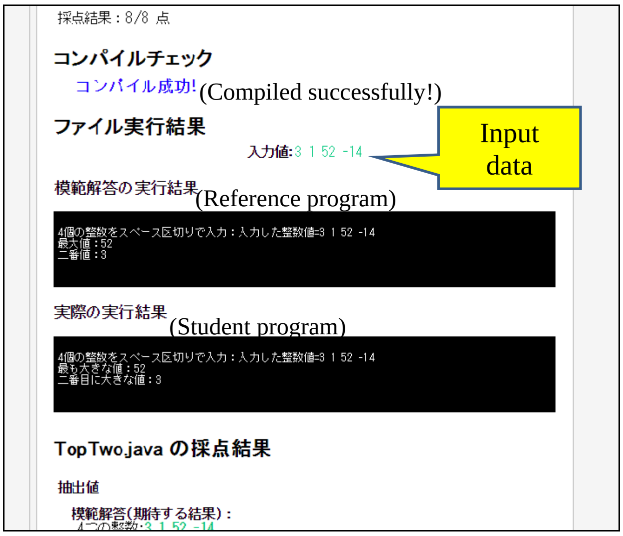
3) Result test

At the end of testing process, our system executes the compiled class files giving test data, and saves the result as a text file in the “results” directory. It folks a sub-process and executes a program by calling command prompt. In the command prompt, as described previously, the system uses redirection from the console input to text files prepared in advance.

If no error occurs, the system executes the reference program in the same way, and compares the results with the saved results of the student program. Basically, if the two results are equivalent, the student program will be scored as a correct answer. There may be a student who writes a program that displays always the same results without any computation with the input data. Therefore, our system uses two or more input data not to be cheated by such programs. If the assignment demands to use random numbers for the computation, the system checks only the data format.

E. Displaying Results

As the result of the above procedures, the system displays a result page as shown in Fig. 6. In this example, since the program is composed of only one main method, the system omits the JUnit test.



採点結果：8/8 点

コンパイルチェック

コンパイル成功!(Compiled successfully!)

ファイル実行結果

入力値:3 1 52 -14

模範解答の実行結果(Reference program)

```
4個の整数をスペース区切りで入力：入力した整数値=3 1 52 -14
最大値：52
二番値：3
```

実際の実行結果(Student program)

```
4個の整数をスペース区切りで入力：入力した整数値=3 1 52 -14
最も大きな値：52
二番目に大きな値：3
```

TopTwo.java の採点結果

抽出値

模範解答(期待する結果)：

4つの整数:3 1 52 -14

Fig. 6. Result page.

If there were an error the detail of compiler error would be displayed. Since the compiler check is passed, the executed result is displayed on the bottom with the executed result of the reference program is displayed. Thus, the student gets useful information for correction if his/her program failed one of the tests.

VI. Evaluation

A. Preliminary Experiment

We evaluated our system in a batch mode to find errors and improve the accuracy of scoring student programs. To run the evaluation, we used archived real student programs that are submitted for assignments used in the courses of 2010 and 2013.

First, we confirmed that the Java Compiler API woks similarly to javac command in the command prompt. Next, we checked the character encoding of source files, because student programs are required to output Japanese characters. Both Shift-JIS and UTF-8 are common encoding for Japanese characters. In order to avoid unnecessary troubles, we forced students to use only Shift-JIS.

One of the most serious troubles found by the evaluation was in the result test phase. As mentioned before, the regular expressions are used to compare the result text of student programs and the reference program. When the text straddles over multiple lines, the matching may fail. In addition, when the text includes a floating point number, the decimal fraction may differ. We improved the regular expressions to extinguish such errors.

B. Live Experiment

1) Accuracy evaluation

We finally introduced our system to the actual course in 2014 and imposed 6 assignments shown in Table I. Table II shows the accuracy of scoring results of the system.

TABLE II: Accuracy of Scoring

ID	Files	Correct	Incorrect	Accuracy
1	134	132	2	98%
2	124	119	5	96%
3	120	120	0	100%
4	111	110	1	99%
5	111	108	3	97%
6	106	106	0	100%
Total	706	695	11	98%

As shown in this table, the overall accuracy is very satisfactory.

2) Performance evaluation

We measured the response time of our system. The response time is defined as elapsed time from receiving a request on the server to sending out the scoring results to the browser. As the nature of Tomcat, when the servlet is accessed at first time, it takes more time to generate an instance. Therefore, we measured the first and second requests for a total of 60 times in all the assignments.

As a result, the average time of response for every assignment was 0.99 seconds for the first time access, and 0.6 seconds for the second time access. This demonstrates that the system works effectively enough for returning a feedback to a student immediately.

3) Questionnaire survey

TABLE III: Questionnaire List

No.	Question
Q1	Was the result of automatic scoring correct?
Q2	Was the result easy to understand?
Q3	Was this system user-friendly? (login, submission, etc.)
Q4	Was the automatic scoring function useful to solve programming assignments?

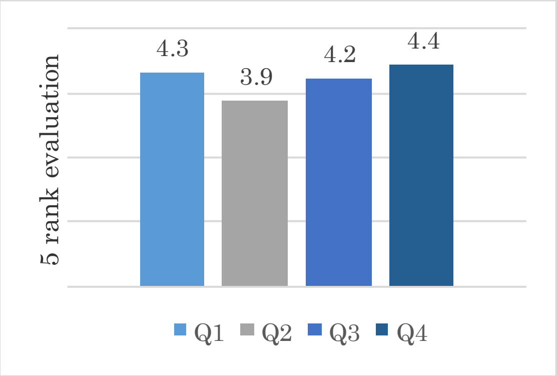


Fig. 7. Survey result.

We performed a questionnaire survey about the system at the end of the course. Students evaluated the following items in five ranks (5: Highest, 1: Lowest); 1) Accuracy of scoring, 2) Understandability of the result, 3) Usability of the system, 4) Usefulness for programming assignments. The questionnaire list is shown in table III.

Fig. 7 shows the result. Students totally gave good evaluation results. Especially, the average rank of the usefulness was 4.4.

Moreover, several students said "Resubmission was helpful," "I could find mistakes that I could not find by myself," "I could check the behavior of my program anytime anywhere." We can conclude that our system is very effective for learning a programming language.

VII. Conclusion

This paper has proposed a web-based automatic scoring system for Java programming assignments. The system tests student programs in three phases: compiler check, Junit test, and result test. For testing private classes and methods typically found in the elementary course, the Java Reflect API is used in the JUnit test. Moreover, to compare the output text of student programs and the reference program, the regular expressions are used in the result test. The system was introduced to the actual programming course in 2014, and showed very good accuracy and performance. Both of the instructor and students have enjoyed teaching and learning Java by the system.

Current version of the system requires the instructor to write comparator programs using the regular expressions in Java for the result test phase. We are planning to develop an automatic comparator generator to further reduce the burden of the instructor. Error messages of the compiler are not easily understandable for novice students. Transforming these messages into natural Japanese language is required. Finding plagiarisms is another important issue [7], [8]. Integration with a plagiarism finder is future work.

References

[1] C. Beierle, M. Kula, and M. Widera, "Automatic analysis of programming assignments," in *Proc. Der 1. e-Learning Fachtagung Informatik (DeLFI)*, 2003, pp. 144-153.

[2] J. C. Dutton. (2001). WebAssign: A better homework delivery tool. [Online]. Available: http://technologysource.org/article/webassign/

[3] J. L. F. Alemán, "Automated assessment in a programming tools course," *IEEE Tran. on Education*, vol. 54, no. 4, pp. 576-581, 2011.

[4] R. Singh, S. Gulwani, and A. Solar-Lezama. "Automated feedback generation for introductory programming assignments," in *Proc. 34th ACM SIGPLAN conf. on Programming Language Design and Implementation (PLDI)*, 2013, pp. 15-26.

[5] J. Li, W. Pan, R. Zhang, F. Chen, S. Nie, and X. He, "Design and implementation of semantic matching based automatic scoring system for C programming language," *Lecture Notes in Computer Science (LNCS)*, vol. 6249, 2010, pp. 247-257.

[6] F. Shamsi and A. Elnagar, "An intelligent assessment tool for students' Java submissions in introductory programming courses," *Journal of Intelligent Learning Systems and Applications (JILSA)*, vol. 4 no. 1, pp. 59-69, 2012.

[7] U. Garg, "Plagiarism and detection tools: An overview," *Research Cell: Int. Journal of Engineering Sciences*, vol. 2, pp. 92-97, 2011.

[8] U. Inoue, "Finding plagiarisms with evidence in Java programming courses," *Int. Journal of Computer Theory and Engineering (IJCTE)*, vol. 6, no. 2, pp. 91-96, 2014.

Hiroki Kitaya received his B.Eng. degree in information and communication engineering from Tokyo Denki University, Japan, in 2013. He is an M.Eng student of the same university. His technical interests include web systems and programming languages.

Ushio Inoue received his B.Eng. and D.Eng. degrees from Nagoya University, Japan in 1975 and 1992 respectively. In 1975, he joined NTT Laboratories, where he was engaged in research of information retrieval and database management systems. In 1995, he was a senior research manager at NTT Data, where he developed distributed and multimedia information systems. Since 2004, he is a professor of Tokyo Denki University, Japan. Current his research interests include geographic information systems, information retrieval and recommendation, and education support systems. Prof. Inoue is a member of ACM, IEICE, IPSJ, and GISA. He is currently a vice chair of ACM SIGMOD Japan.