ICIC Express Letters
Part B: Applications
Volume **14**, Number **12**, December **2023**
 ISSN 2185-2766
pp. **1235–1241**

A PROGRAMMING LEARNING SYSTEM BASED ON MATHEMATICAL FIGURES

Genki Morikawa, Shunya Hashi, Katsuya Masui and Hiroshi Kamada*

Department of Media Informatics
College of Informatics and Human Communication
Kanazawa Institute of Technology
3-1 Yatsukaho, Hakusan, Ishikawa 924-0838, Japan
{ b1939983; b1918920; b1906925 }@planet.kanazawa-it.ac.jp
*Corresponding author: kamada@neptune.kanazawa-it.ac.jp

Received March 2023; accepted June 2023

Abstract. *Currently, the supply of programmers is not keeping up with the demand for programmers, which is increasing year by year due to the progress of IT in our society. Therefore, in order to develop IT human resources, we created a programming learning system for students with no programming experience to maintain their interest in programming by using mathematical figures as the subject matter. In order to solve the problem of the conventional system's inability to develop coding skills, this system includes, in addition to the multiple-choice questions, descriptive questions in which students are asked to draw 2D and 3D figures using programming. Since the open-ended questions can only be tackled with a solid understanding of the essence of programming, this system has been designed to provide students with a deeper understanding of programming as well as the fundamentals. In addition, in order to accurately identify programming errors and sustain learning, the system has a correct/incorrect judgment and advice function according to the errors. In the evaluation experiment, the system received a high evaluation for knowledge acquisition, and it was confirmed that the direction of this system was generally appropriate.*
Keywords: Programming, Learning systems, Math shapes, Advisory functions

1. **Introduction.** In recent years, the shift to IT in the Fourth Industrial Revolution has increased the importance of programmers who can cope with this trend. Based on our own analysis of statistical data from various countries, we know that Japan ranks 4th in the world in the number of IT engineers, with 1.09 million, but only 27th in the growth rate of IT engineers [1]. In addition, as shown in Figure 1, the number of people employed as information processing and telecommunications engineers has been rising steadily from 2010 to 2017, indicating that the demand for IT personnel is increasing [2]. On the other hand, the supply of human resources for programmers has not kept pace due to factors such as the declining birthrate and aging population. Accordingly, the development of human resources has begun, with programming classes in elementary schools starting in 2020. However, programming learning has unique difficulties [3], making it difficult to continue learning. Now that there are more opportunities to be exposed to programming, the development of a learning system that encourages continuous programming learning is desired. A system using a gamification framework [4] exists as a learning system that encourages continuous learning, and it has been shown to be effective in promoting continuous learning. There are also systems that introduce a function that offers advice for wrong answers and a function that allows the user to write and execute programming [5,6]. However, these systems are almost equal to general materials in terms of learning subject matter, and it is difficult to say that they are effective in sustaining programming

DOI: 10.24507/icicelb.14.12.1235

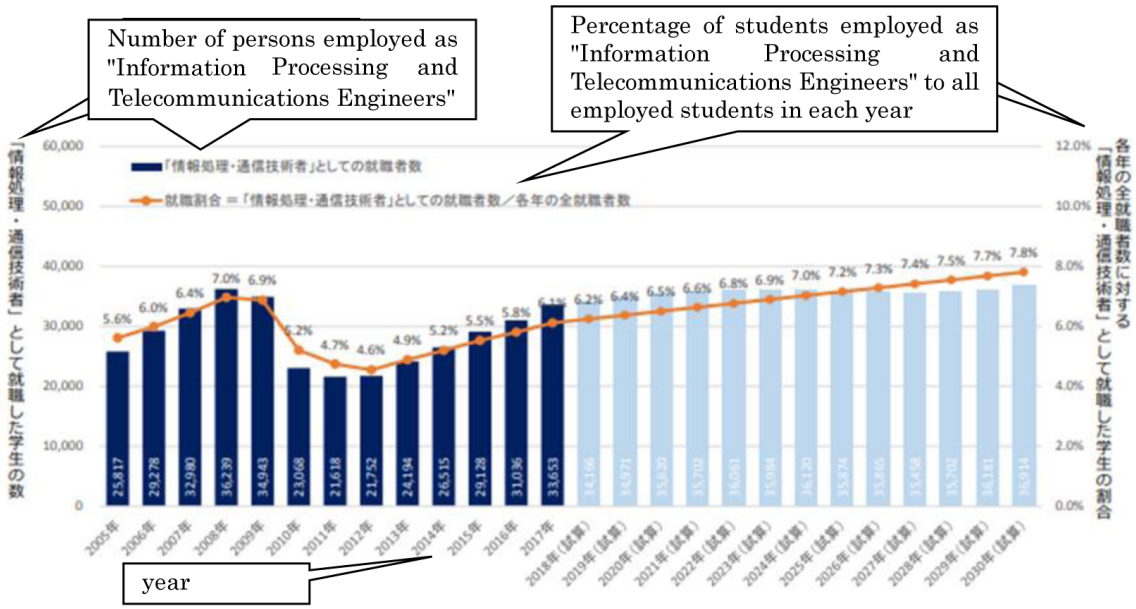

FIGURE 1. Number of people employed as "information processing and telecommunications engineers" and percentage employed as IT personnel [2]

learning, even for those who do not want to learn or are new to it. To solve the problem, the system must be one that beginners want to continue learning. There is also a programming learning system for children [7], but while this system is easy to understand and enables continuous learning, it has the problem that it does not allow developmental learning. The use of interactive artificial intelligence GPT in education is also limited because program bug resolution using GPT is limited [8]. Based on this situation, we have created a learning system that will serve as a starting point for programming for students of all ages who have never been exposed to programming as part of our efforts to nurture future programmers.

Conventional systems had problems that they did not attract interest and develop coding skills and application skills. To solve these problems, we implemented descriptive questions and two types of questions, 2D and 3D. As a result, we were able to achieve an appropriate level of difficulty that leads to the acquisition of programming knowledge. In particular, high evaluations were obtained for "difficulty of the question", "knowledge acquisition" and "sustained interest".

This paper is organized as follows: Section 2 describes the conventional learning systems and the problems, Section 3 describes the solution in this system, Section 4 describes the functions of this system, Section 5 describes the evaluation results and discussion of this system, and finally Section 6 provides a summary of this study.

2. **Conventional Systems and Problems.** A typical example of a conventional system is the "elementary school math programming materials" [9] shown in Figure 2. This teaching material was created by an educational publisher to provide elementary school students with an introduction to programming. As shown in Figure 2, the system consists of drag-and-drop instruction blocks such as "go forward" and "turn" to move a car and trace a shape by its trajectory. The operation is simple and easy to understand, and allows users to understand the basics of programming. Another conventional study is the "a visual code programming learning system for children" [7]. The system uses optical illusions to attract children's interest while they learn programming, and then checks the results of their learning with multiple-choice questions. The system uses colors, shapes, and animations to explain functions, making learning visually enjoyable. One problem with

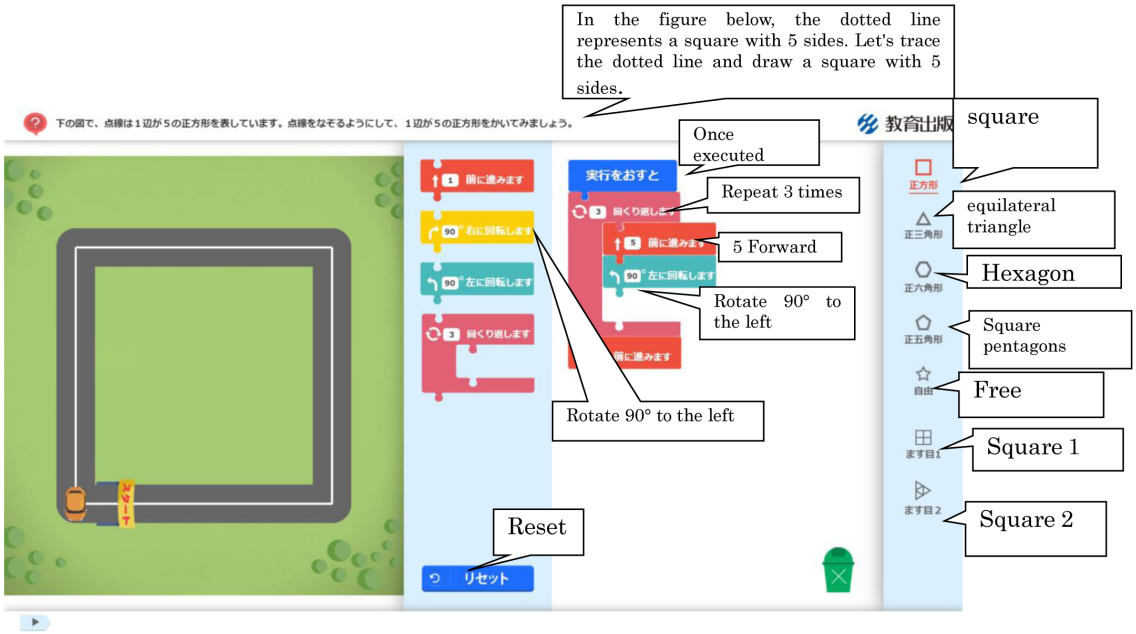

FIGURE 2. An operation screen example [9]

these systems is that they do not develop the coding ability to freely write programs, since the students are not actually programming, only arranging instruction blocks or selecting options. Next, we will describe about conventional studies, "a description-type programming learning system which advises about main wrong answers" [5] and "a learning system for object-oriented programming themed on color" [6]. There are both multiple-choice and descriptive questions, and the user answers the questions by selecting options or writing program sentences according to the question. The system then determines whether the answer is correct or incorrect, and the user learns programming based on the advice displayed according to the answer. One of the problems with these systems is that they only offer problems for flat surfaces, and therefore, the user cannot learn how to program to create three-dimensional objects, and thus lacks the ability to apply the skills. In summary, conventional systems had problems that they did not attract interest and develop coding skills and application skills.

3. **Solution in This System.** For the problem that coding ability cannot be cultivated, the method of having students actually write programs was used to help develop coding skills. In addition, the addition of a correct/incorrect judgment function to determine if the programming answer is correct is also used to develop coding skills. For problems that systems do not attract interest and cannot develop application skills, we provided a question to create a complex two-dimensional figure and a question to code a three-dimensional figure, allowing the respondent to think from multiple perspectives. Examples of 3D problems are shown in Figures 3 and 4. Figure 4 is a Processing program that draws the quadrilateral in Figure 3. After specifying the vertices and drawing, the image of the display window is saved to a file. At first glance, the problem looks simple, but it is a complex problem that requires specifying the coordinates of all vertices, drawing a plane, and then combining the planes to draw an arbitrary polygon, which requires a certain degree of ingenuity to develop the respondent's applied skills. In addition, by incorporating familiar mathematical figures into the problems, we have tried to make them more interesting to work on.

4. **The System.** This system used Processing as the learning language; Processing is a language for learning coding skills, specialized for graphic functions [10]. Since the results

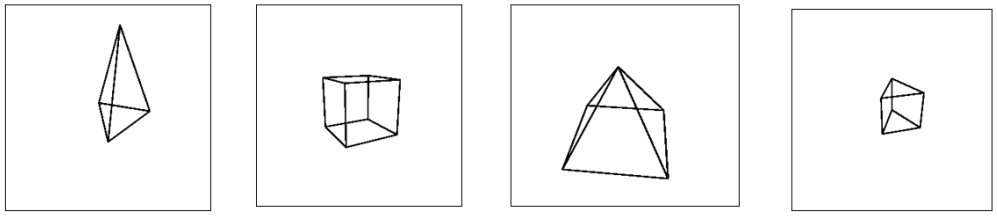

FIGURE 3. The 3D problem of this system

```
void draw(){
  translate(width/3, height/2, 0);
  rotateX(0.8);
  rotateZ(0.1);
  beginShape(TRIANGLE_FAN);
  vertex(50, 50, 100);
  vertex(0, 0, 0);
  vertex(0, 100, 0);
  vertex(100, 100, 0);
  vertex(100, 0, 0);
  vertex(0, 0, 0);
  endShape();
  saveFrame("hikaku/data/img.png");
}
```

FIGURE 4. Example of 3D description problem

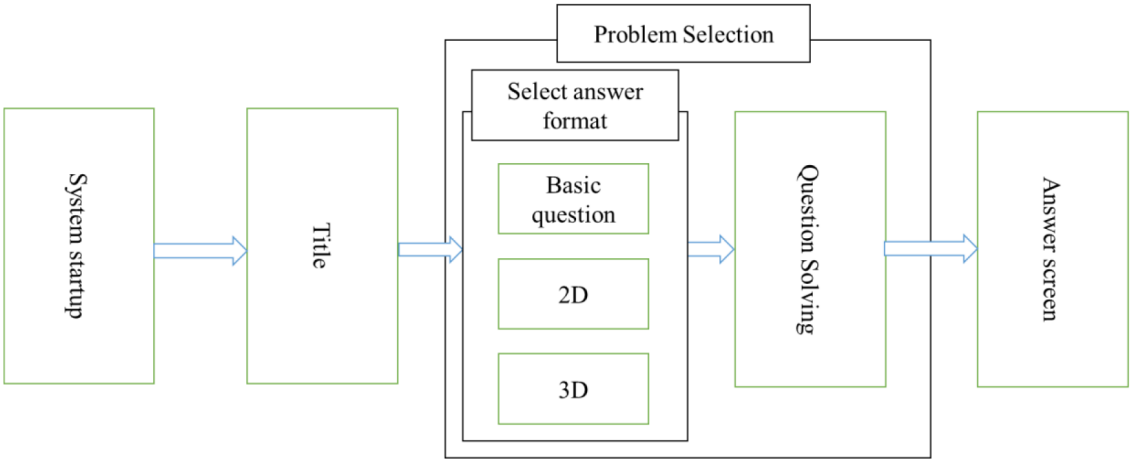

FIGURE 5. Screen transitions from system startup to the answer screen

of programming are immediately fed back visually, errors are easy to understand visually, and the system is expected to be useful for judging correctness and presenting advice through images. In addition, since Processing is a simplified version of Java, it is easy for programming beginners to get used to, and is considered suitable as a foundation for learning other languages as a next step. After starting the system, the title screen is displayed, and the "Start" button is clicked to proceed through the system. After clicking the "Start" button, you will see a button to select the answer format, where you can choose either the multiple-choice question or the essay question. After selecting the answer format, select the questions. There are 16 basic questions, 16 2D questions, and 4 3D questions. The screen transition from system startup to answering is shown in Figure 5.

Two types of questions were implemented in this system: selective and descriptive. Both types of questions are about writing a program to draw a figure. Compared to the descriptive type, the selective type is less difficult and can be tackled relatively easily by

learners who are not good at programming or who are new to programming in the first place. The Short Answer type is more difficult than the Multiple Choice type because it does not provide a list of possible answers like the Multiple Choice type, but it can be considered as a way to acquire practical coding skills because of the actual programming process. There are 16 questions in total (8 questions × 2 pages), and the question screen appears when you click on the question column. The questions are in the form of fill-in-the-blanks questions with three choices, one of which is the correct answer. To answer a question, left-click on each choice. When the correct answer is selected, a message "Correct answer!" and the template of the function used for the answer is displayed. When an incorrect answer is selected, the words "Too bad..." and a clue corresponding to the question will be displayed. The descriptive questions are divided into 2D and 3D questions. 2D questions are 4 questions × 4 pages, for a total of 16 questions, and 3D questions are 4 questions × 1 page, for a total of 4 questions. The 2D questions range from relatively easy ones that use functions to draw figures to more difficult ones that use mathematical formulas to draw figures. The 3D problems are those that draw figures by specifying coordinates and require a certain degree of ingenuity, thus cultivating the solver's ability to apply programming skills. After programming is completed, left-clicking the Run button outputs the result as an image, which is used for the correct/incorrect decision described below. The flow of the solution is shown in Figure 6.

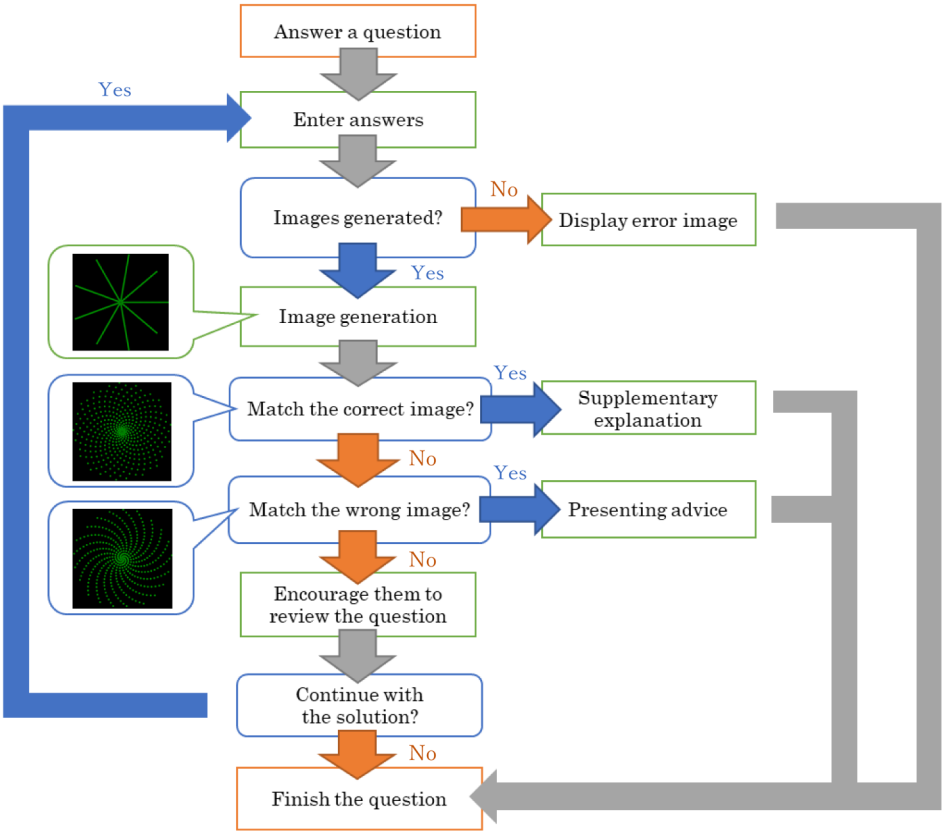

FIGURE 6. Flow of solution

Correctness is determined by comparing the pixel values of the output image and the correct and incorrect images of the data group one by one. It is determined that the images match when all the pixel values are equal, comparing 800px width × 800px height = 640000px for 2D questions and 300px width × 300px height = 90000px for 3D questions. If the output image matches the correct answer image, it is judged as "OK" and if it matches the wrong answer image, it is judged as "NG". If it is judged as "NG", the system provides the learner with advice on how to modify the solution program. The

advice is given by displaying a text string from a text file associated with the wrong answer image stored in the data group after the answer. When an answer that does not match the image in the data set is output, the output image is automatically stored in the data set and used as a new wrong answer. By adding advisory text data to these new wrong answers and reflecting them in the system, the number of wrong answer patterns can be increased. However, the addition of advisory text must be done manually by checking the images. Figure 7 shows an example of the screen of the correct/incorrect judgment function. As shown, the output image is displayed in the upper left corner of the screen, the correct answer image is displayed in the upper right corner of the screen, and the advice text is displayed at the bottom of the screen. In the case of the example in Figure 7, there is an error in the formula in the program, which is due to the fact that the error is caused by not adding ✓to the formula, so the advice is given on this point.

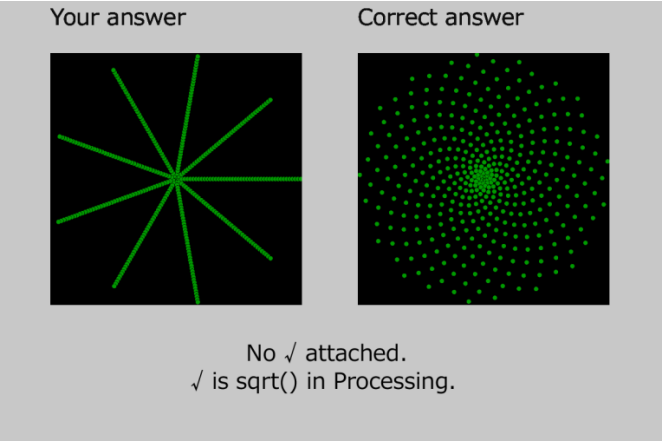

FIGURE 7. Example of the correctness screen

5. **Evaluation Experiment and Discussion.** Seventeen students were asked to rate the system introduced so far on a 5-point scale, with 1 being the lowest and 5 being the highest. The five items on the questionnaire were "difficulty of the question", "knowledge acquisition", "sustained interest", "novelty", and "usability". The results of this evaluation experiment are shown in Figure 8.

The questionnaire result for "difficulty of the question" in Figure 8 has the highest average value compared to the other items. This may be due to the fact that a wide variety

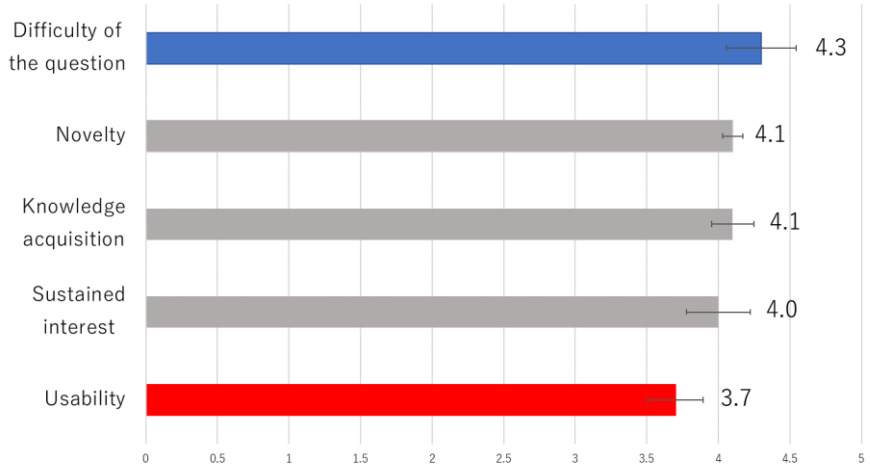

FIGURE 8. Results of evaluation experiment

of questions, from basic to advanced, were included in the survey, and thus each user felt that the level of difficulty was appropriate for him or her. The items of "sustained interest", "knowledge acquisition", and "novelty" each had high values. First, "sustained interest" is thought to have been achieved by incorporating a large number of questions with creative shapes and colors of figures, so that users would not feel bored. Secondly, as for "knowledge acquisition", the use of fill-in-the-blanks and descriptive questions made it possible for users who had never touched the program to understand it step-by-step. Lastly, the "novelty" of the system is considered to be high because the use of mathematical figures as the subject matter enables the realization of a seemingly infinite number of patterns as a programming learning system. The reason why poor (level 2) was given for "usability" is probably due to the fact that the general specifications of this system have not been changed since its creation. We also felt that the system did not take into account that users may not be familiar with the operation of personal computers.

6. **Conclusion.** We have created a programming learning system that incorporates mathematical fields so that people who have never touched programming can easily become familiar with it. The system incorporates both fill-in-the-blanks and fully descriptive questions, so that users can learn programming more efficiently by working on questions that match their level of difficulty. In addition, by using mathematical figures as the subject matter, the system is easy to use for programming beginners, as they can easily visualize the finished product when actually creating it.

With the exception of a few items, this system received a high evaluation. However, the system does not seem to have given much consideration to ease of use. We felt that we would like to have the users actually experience the programming by looking at the programming screen, but in order to ensure that people who see the programming system for the first time can use it without inconvenience, we need to devise a way to explain the system's operation and incorporate descriptive questions into the screen like fill-in-the-blanks questions.

Acknowledgments. This work was supported by JSPS KAKENHI Grant No. 22K02869.

REFERENCES

[1] Human Resocia, *[Human Resocia Research] Global IT Engineer Report Vol.1 92 Countries*, 2020, https://git.resocia.jp/en/info/post-developers-around-the-globe-survey/, Retrieved on February 22, 2022.

[2] Mizuho Information & Research Institute, Inc., *Survey on IT Human Resource Supply and Demand*, Survey Report, 2019, https://www.meti.go.jp/policy/it_policy/jinzai/houkokusyo.pdf, Retrieved on February 22, 2022 (in Japanese).

[3] M. Sivasakthi and R. Rajendran, Learning difficulties of 'object-oriented programming paradigm using Java', *Indian Journal of Science and Technology*, vol.4, no.8, pp.983-985, 2011.

[4] L. Christopher and A. Waworuntu, Java programming language learning application based on octalysis gamification framework, *International Journal of New Media Technology*, vol.8, no.1, pp.65-69, 2021.

[5] S. Sasaki and H. Kamada, A description-type programming learning system which advises about main wrong answers, *ICIC Express Letters, Part B: Applications*, vol.12, no.9, pp.807-813, 2021.

[6] R. Moto, D. Horii and H. Kamada, A learning system for object-oriented programming themed on color, *ICIC Express Letters, Part B: Applications*, vol.12, no.9, pp.815-822, 2021.

[7] T. Hasegawa, S. Yamaguchi and H. Kamada, A visual code programming learning system for children, *ICIC Express Letters, Part B: Applications*, vol.12, no.10, pp.909-916, 2021.

[8] N. M. S. Surameery and M. Y. Shakor, Use Chat GPT to solve programming bugs, *International Journal of Information Technology and Computer Engineering*, vol.3, no.1, pp.17-22, 2023.

[9] KYOIKU-SHUPPAN Co., Ltd., *Elementary School Math Programming Materials*, 2020, https://www.kyoiku-shuppan.co.jp/docs/sansu/programing/index.html, Retrieved on February 22, 2022 (in Japanese).

[10] Processing.org, *Processing*, https://processing.org/, Retrieved on February 22, 2022.