International Journal of Sciences: Basic and Applied Research (IJSBAR)

ISSN 2307-4531
(Print & Online)

http://gssrr.org/index.php?journal=JournalOfBasicAndApplied

The Class Practice for Advancing from CS Unplugged to Full-Fledged Programming

Makoto Tanabe [a], Tatsuhiro Tamaki [b], Atsushi Onishi [c], Makoto Sakamoto [d], Yasuo Uchida [e]*

[a, e] *National Institute of Technology, Ube College, 2-14-1, Tokiwadai, Ube 755-8555, Japan*

[b] *National Institute of Technology, Okinawa College, 905, Henoko, Nago-shi 905-2192, Okinawa, Japan*

[c] *National Institute of Technology, Tsuyama College, 624-1, Numa, Tsuyama 708-8509, Japan*

[d] *University of Miyazaki, 1-1, Gakuen Kibanadai-nishi, Miyazaki 889-2192, Japan*

[a] *Email: tanabe@ube-k.ac.jp*

[b] *Email: t.tamaki@okinawa-ct.ac.jp*

[c] *Email: a-onishi@tsuyama-ct.ac.jp*

[d] *Email: sakamoto@cs.miyazaki-u.ac.jp*

[e] *Email: uchida@ube-k.ac.jp*

Abstract

The course we teach has about 40 students per class, aged 16 through 17 years. Since the students' major field of study is Management Information, they need to learn programming techniques. However, the results of the survey described below show that not a few students consider themselves to have insufficient understanding of programming or think that they are not good at programming. We are examining ways to improve this situation. CS Unplugged is a method of teaching information science without using computers, first proposed by Tim Bell of the University of Canterbury in New Zealand. While CS Unplugged is said to be effective in teaching information science. To address this topic, we implemented CS Unplugged for a group of students, and looked at their responses. Accordingly, we propose a new method of advancing from CS Unplugged to full-fledged programming. The proposed method begins with conducting a CS Unplugged activity, and then continues on to writing a program on the same theme and further to its abstraction in Java. Accordingly, we propose advancing from CS Unplugged to full-fledged programming through a new Four-Step Method.

---

* Corresponding author.

The proposed method consists of the following steps: Step 1, A CS Unplugged activity; Step 2, A CS Plugged activity; Step 3, Preparing pseudocode; and Step 4, Writing Java source code.

Keywords: CS Unplugged; CS Plugged; Java; Programming.

1. Introduction

The course we teach has about 40 students per class, aged 16 through 17 years. Since the students' major field of study is Management Information, they need to learn programming [1, 2] techniques. However, the results of the survey described below show that not a few students consider themselves to have insufficient understanding of programming or think that they are not good at programming. We are examining ways to improve this situation. CS Unplugged [3] is a method of teaching information science without using computers, first proposed by Tim Bell of the University of Canterbury in New Zealand. While CS Unplugged is said to be effective in teaching information science [4], there have been concerns that its success or failure may be an effect of the skill and experience of instructors. To address this topic, we implemented CS Unplugged for a group of students of a different age group, from fifth through ninth grades, and looked at their responses. The results showed that after first making sufficient preparations the method could generate results without necessarily depending on the skill or experience of instructors. Other research underway in Japan includes a study on use of teaching aids in learning about algorithms through CS Unplugged [5] and a study on learning the fundamentals of computer programming through a programming learning environment for beginners [6]. A study concerns practice in teaching high-school students [7], and it has been reported to have had some, albeit limited, success.

However, at present almost no research has been conducted on advancement from CS Unplugged to full-fledged programming languages. Accordingly, we propose a new method of advancing from CS Unplugged to full-fledged programming. The proposed method begins with conducting a CS Unplugged activity, and then continues on to writing a program on the same theme and further to its abstraction in Java. Accordingly, we propose advancing from CS Unplugged to full-fledged programming through a new Four-Step Method. The proposed method consists of the following steps: Step 1, A CS Unplugged activity; Step 2, A CS Plugged activity; Step 3, Preparing pseudocode; and Step 4, Writing Java source code.

2. Background of this study

Every year we conduct a survey following our first-term course of two 90-minute sessions per week held over 30 weeks. Here we will look at numbers of students who answered "yes" or "no" to this survey's question, "Do you think you largely understand Java?"

As seen in Figure 1, the survey's results show that not a few students consider themselves to have insufficient understanding of programming or think that they are not good at programming. As such, there is a need to improve this situation.

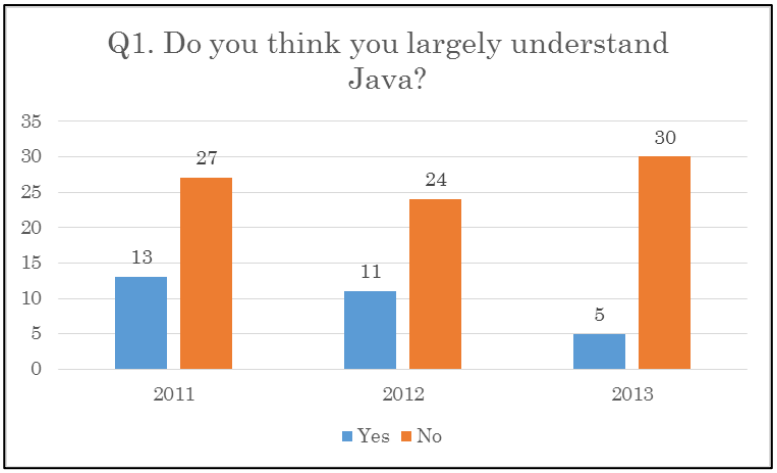

Figure 1: Survey results on understanding of Java

3. The Four-Step Method

In this paper, we use the name CS Plugged [8] to refer to implementing a CS Unplugged activity through a computer program. The goal is to advance to computer programming through using a computer program to conduct the work done by human beings in a CS Unplugged activity.

Accordingly, we propose advancing from CS Unplugged to full-fledged programming through a Four-Step Method. The proposed method consists of the following steps: Step 1, A CS Unplugged activity; Step 2, A CS Plugged activity; Step 3, Preparing pseudocode; and Step 4, Writing Java source code.

3.1. Step 1: CS Unplugged

The activity we implemented was CS Unplugged Image Representation activity. Figure 2 shows a scene from this activity.

Figure 2: Scene from course practice

3.2. Step 2: CS Plugged

Here we will look at the example of using a computer program to implement the CS Unplugged activity Image Representation. This is conducted through two activities. The first, converting the image to code, is represented in Figure 3. When students click on squares in the grid at left to draw a picture, the image is converted instantly to code as displayed at right.

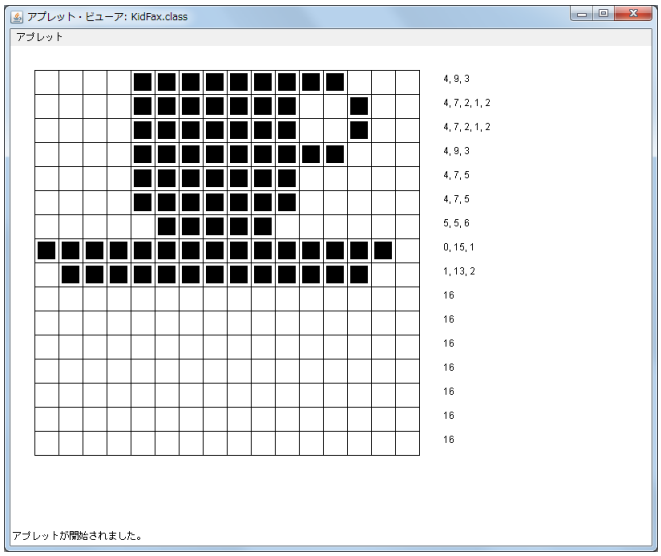

Figure 3: Applet for converting image to code.

The other activity is the reverse of this process, with students reproducing a picture from code (Figure 4). They enter code to the text boxes at right in a format such as “6, 5, 2, 3” and press the Enter key to display the resulting image instantly in the grid at left.

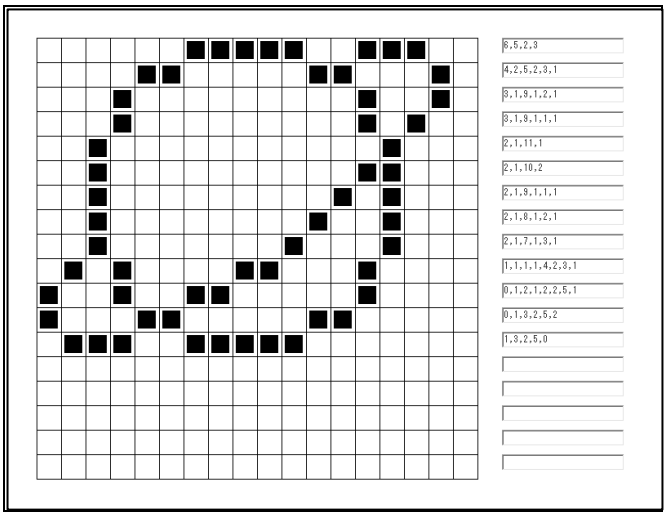

Figure 4: Applet for reproducing an image

3.3. Step 3: Preparing pseudocode

In this step, the three elements of sequence, decision, and repetition through preparation of the trace table in the preceding step are extracted and represented in pseudo-language. In Japan, the national Information Technology Engineer Examinations employ pseudo-language [9]. A pseudo-language simulator is a type of software that makes such pseudo-language executable. In this paper, we used the freeware pseudo-language simulator SARA [10]. Figure 5 shows the code written in this step and its execution.

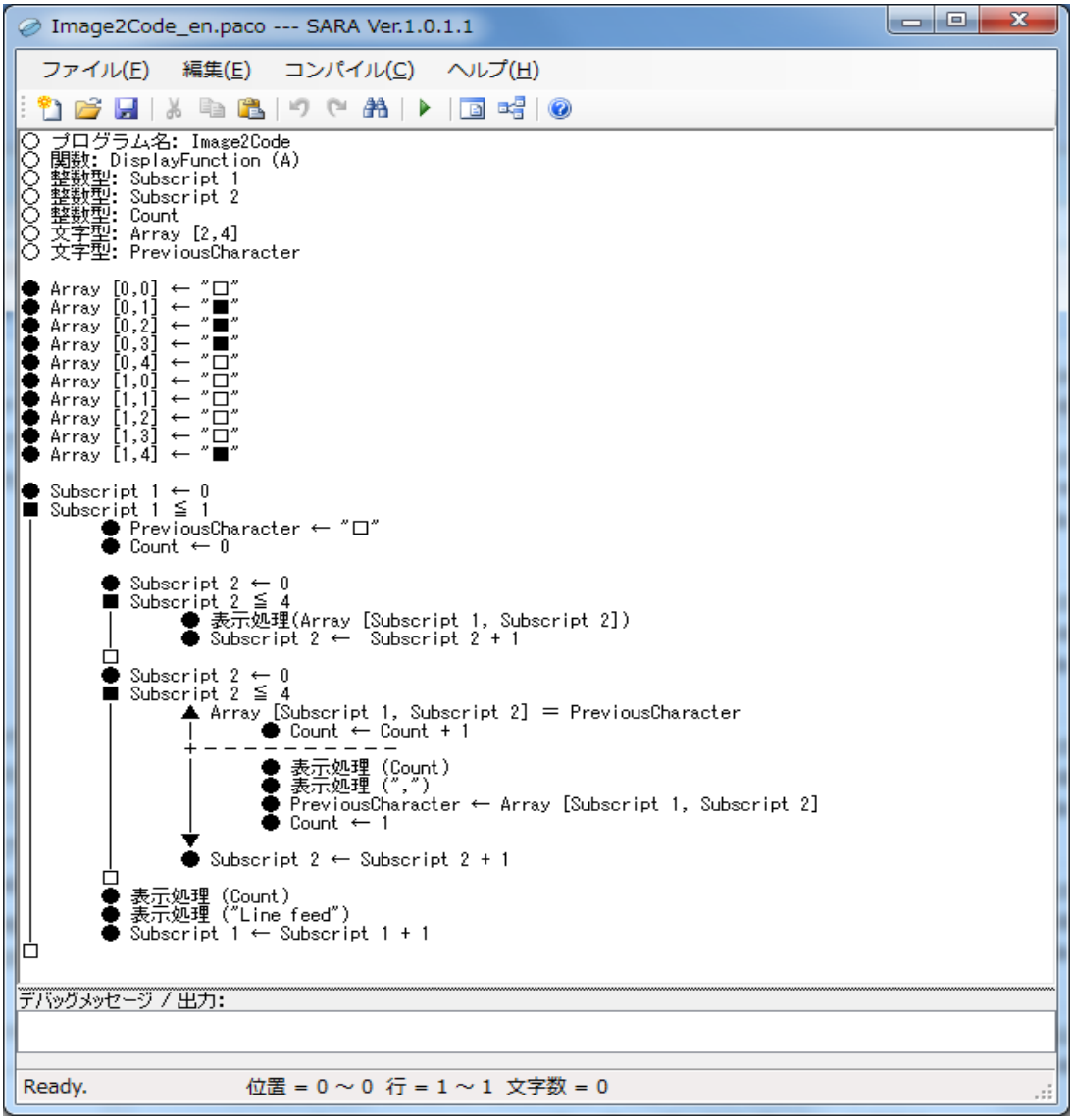

Figure 5: Sample use of pseudo-language simulator

3.4. Step 4: Writing Java source code

In this step students create a program by converting the pseudo-language from the previous step to Java source code (Figure 7). The subsequent debugging process is handled by going back and examining each previous step depending on the content of the error messages.

```
public class Image2Code {
public static void main(String args[]) {
char[][] image = {{'□', '■', '■', '■', '□'},
{'□', '□', '□', '□', '■'},
{'□', '■', '■', '■', '■'},
{'■', '□', '□', '□', '■'},
{'■', '□', '□', '□', '■'},
{'□', '■', '■', '■', '■'}
};
// Iterate only number of lines in an array
for(int i = 0; i < image.length; i++) {
char previous = '□';
int count = 0;
// Display the image in pixels
for(int j = 0; j < image[0].length; j++) {
System.out.print(image[i][j]);
}System.out.print(" ");
// Count number of adjoining pixels of same color and display in numerical form (code)
for(int j = 0; j < image[0].length; j++) {
if (image[i][j] == previous) {
count++;
} else {
previous = image[i][j];
System.out.print(count + ", ");
count = 1;
}
}
System.out.println(count);
}
}
}
```

Figure 7: Java source code

```
□■■■□ 1, 3, 1
□□□□■ 4, 1
□■■■■ 1, 4
■□□□■ 0, 1, 3, 1
■□□□■ 0, 1, 3, 1
□■■■■ 1, 4
```

Figure 8: Results of running the above program

4. Practice Survey Result

On July 16, 2015, we surveyed second-year students in the Department of Management Information on CS Unplugged and CS Plugged (collecting replies anonymously).

- **Questionnaire format**

Circle the number corresponding to the correct answer for each of the questions below.

Q1. Do you think CS Unplugged (manual) helped you to learn programming?

Q2. Do you think CS Plugged (manually checking program behavior) helped you to learn programming?

1. Yes 2. Somewhat 3. Not much 4. No

Note: If possible, please write the reasons for your choices above.

- **Consideration of survey results**

Table 1: Survey results

Date	Class	Students	Q1	Q2
July 16, 2015	2B	42	A1:17 (40%) A2:21 (50%) A3:4 (10%) A4:0 (0%) N/A:1 (2%)	A1:21 (51%) A2:17 (41%) A3:3 (7%) A4:0 (0%) N/A:0 (0%)

Q1: Results showed that 40% of respondents answered "Yes," 50% "Somewhat," 10% "Not much," and 0% "No." Overall, the majority of responses were favorable.

Q2: Results showed that 51% of respondents answered "Yes," 41% "Somewhat," 7% "Not much," and 0% "No." Overall, the majority of responses were favorable.

5. Conclusion

We have proposed a new method of advancing from CS Unplugged through the new process of CS Plugged to full-fledged computer programming languages, as a means of deepening understanding in computer programming education.

As we have shown above, we got favorable responses about Image Representation activity. So we can use this subject as the effective activity in computer programming education from now on. But to make students skilled in programming, we need more other effective activities. To make these activities is our future work.

References

[1] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan et al. “A multi-national, multi-institutional study of assessment of programming skills of first-year CS students” in *Proc. ITiCSE-WGR '01*, 2001, pp. 125-180.

[2] Y. Hofuku, S. Cho, T. Nishida and S. Kanemune. “Why is programming difficult? Proposal for learning programming in "small steps" and a prototype tool for detecting "gaps"” in *Proc. ISSEP 2013*, 2013, pp. 13-24.

[3] The Unplugged Community. “Computer Science Unplugged” Internet: csunplugged.org/, [Oct. 6, 2015].

[4] Y. Idosaka, Y. Kuno and S. Kanemune. “Attempt and the Practice of Class Method Improvement Based on "Computer Science Unplugged"”, *Journal of the Japan Society of Technology Education*, vol. 53, no. 2, pp. 115-123, Jun. 2011.

[5] H. Manabe, S. Kanemune and M. Namiki. “Effects of Teaching Tools in CSU Algorithm Education”, *IPSJ Journal*, vol. 54, no. 1, pp. 14-23, Jan. 2013.

[6] T. Nishida, A. Harada, R. Nakamura, Y. Miyamoto and T. Matsuura. “Implementation and Evaluation of PEN: The Programming Environment for Novices”, *IPSJ Journal*, vol. 48, no.8 pp. 2736-2747, Aug. 2007.

[7] Y. Feastery, L. Segarsz, S. K. Wahbay and J. O. Hallstrom. “Teaching CS Unplugged in the High School (with Limited Success)” in Proc. *16th annual joint conference on Innovation and technology in computer science education*, 2011, pp. 248-252.

[8] M. Tanabe, Y. Uchida and M. Sakamoto. “A Proposal for Teaching Programming Through the Four-Step Method” *Australian Journal of Basic and Applied Sciences*, 9(14) Special 2015, pp. 1-6, Apr. 2015.

[9] Information-technology Promotion Agency Japan. “Kyotsu ni shiyo sareru gijigengo no kijutsukeishiki (“Symbolic conventions of commonly used pseudo-language”) ” Internet: www.jitec.ipa.go.jp/1_13download /gijigengo_keisiki.pdf, [Oct. 6, 2015].

[10] S. Mimura. “Gijigengo shimyureta SARA (“SARA pseudo-language simulator”)” Internet: mimumimu.net/software/#sara, [Oct. 6, 2015].