

# A TRIAL OF BIDIRECTIONAL TRANSCRIPTION LEARNING FOR BEGINNER PROGRAMMERS

<sup>1</sup>Tatsuhiko Tamaki, <sup>2</sup>Harumi Hashimoto, <sup>3</sup>Atsushi Onishi, <sup>4</sup>\*Yasuo Uchida

<sup>1</sup>National Institute of Technology, Okinawa College, Department of Media Information Engineering, Nago, Japan

<sup>2</sup>Setsunan University, Department of Business Administration and Information, Osaka, Japan

<sup>3</sup>National Institute of Technology, Tsuyama College, Department of Integrated Science and Technology, Tsuyama, Japan

<sup>4</sup>National Institute of Technology, Ube College, Department of Business Administration, Ube, Japan.

\* Corresponding author: E-mail address: uchida@ube-k.ac.jp

**ABSTRACT**—Adoption of programming education has become a global trend. In Japan, the Japan Revitalization Strategy 2016, announced by the Headquarters for Japan's Economic Revitalization in 2016, set forth the aim of making programming education compulsory in primary and secondary education. The purpose of this is to cultivate basic logical thinking skills through programming education, as part of efforts to develop and secure human resources for sparking economic growth. On the other hand, it will likely be necessary to review previously existing programming education in ICT human resources development courses at various types of schools. In the programming education for beginners that we are implementing at a college of technology, there are a considerable percentage of students who feel they are not up to programming. Thus, this study proposes "bidirectional transcription learning" for beginner programmers as an educational method to help strengthen programming education. Based on experience, transcription learning is regarded as effective for mastering programming, but we have conducted a trial to further improve efficiency and deepen understanding, and here we provide an overview and report on our results.

**Keywords**—programming; beginner; transcription learning; bidirectional

## I. INTRODUCTION

The adoption of programming education has become a global trend. For example, the U.S. for example, former President Obama and others have highlighted the need for programming education[1], and steps such as making programming education mandatory from the compulsory education stage have already begun in the U.K. and other countries[2].

In Japan, the Japan Revitalization Strategy 2016, announced by the Headquarters for Japan's Economic Revitalization in 2016, set forth the aim of making programming education compulsory in primary and secondary education[3]. The purpose of this is to cultivate basic logical thinking skills through programming education, as part of efforts to develop and secure human resources for sparking economic growth.

On the other hand, it will likely be necessary to review previously existing programming education in ICT human resources development courses at various types of schools. In the programming education for beginners that we are implementing at a college of technology, there is a considerable percentage of students who feel they are not up to programming. Based on past questionnaires for students and other findings, there are thought to be three main obstacles[4]. First is the process of devising algorithms. The second is the abstraction in mapping to a programming language. Third, is acquiring an image of program operation when these are integrated. In this research, we focus primarily on the second of these processes, i.e., the step of bridging natural language and computer language. Therefore, we propose bidirectional transcription learning for beginner programmers as an educational method for strengthening programming education. Based on experience, transcription learning is regarded as effective for mastering programming[5,6], but we have conducted a trial to further improve efficiency and deepen understanding, and here we provide an overview and report on our results..

## II. LITERATURE REVIEW

Programming is said to be difficult for beginners. Thus, first we shall survey the discussion of obstacles when beginners learn programming.

An international opinion survey of more than 500 students and teachers in multiple countries has found that beginner programmers have difficulties in understanding abstract concepts. There is also research posing and discussing the question "Why is programming difficult?"[8]. In high school classes, "repetition" has been pointed out as a point where beginners tend to stumble. Regarding why beginners find repetition difficult, it has been shown that students easily understand repetition that does not involve variables, but have difficulty understanding repetition that uses variables. In light of these results, there have been efforts to develop tools to support step-by-step understanding by beginner programmers. It is also shown to be effective to analyze gaps in instructional materials, which can be another factor causing beginners to stumble. This has confirmed the effectiveness of a "small steps" approach where new material is incorporated a little at a time when a beginner learns new concepts.

There is also a report on problems and solutions for programming education in programming classes at universities[10]. The problems with conventional programming education are identified and analyzed, and new programming teaching methods are proposed. That is shown to be a programming teaching method using the positive emotions of learners as fuel. That is, immediate understanding at the line level is achieved by using games as subject matter to stimulate interest, and providing an explanation while entering program code in real-time. It is pointed out that using the proposed technique reduces the drop-out problems which frequently occur in programming education. However, a limitation of this teaching method is that it assumes a class size up to about 20 students.

One study offered analysis and proposals from the perspective of cognitive science[11]. First, as problem

presentation, it is pointed out that programming has the peculiar character of looking objectively at human thought processes, but effective learning and educational methods for programming have not been established. The study concludes that examining the variables, arrays, and control structures which are key concepts in programming, and utilizing frameworks of thought that students have already constructed in their heads, and casually use in their daily life, is the easiest teaching method from the standpoint of cognitive science.

Now we shall survey discussions of computational thinking and programming thinking.

A survey has been conducted of information education curriculums, including programming education, in foreign countries[12]. The results showed that, as information education in every country, the core is a computational thinking approach which includes programming education, and learning content is defined with the aim of developing abilities such as abstraction, problem analysis, algorithms, data utilization, evaluation, and collaborative work. The content is similar to programming education, and the steps of the process are described as: giving procedure instructions using robots and puzzles in the lower grades of elementary school, creating programs using a visual language and incorporating branching and iteration in the higher grades of elementary school, and developing programs including multiple data types and modules by using text languages in junior and senior high school.

The term "computational thinking" attracted attention due to an essay by Wing[13]. Wing wrote: "Computer science is not computer programming. Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction." However, "computation thinking" is not clearly defined in Wing's essay. One study has investigated the concept of Computational Thinking (CT)[14].

Programming thinking was discussed at "About the Way of Programming Education in the Elementary School Stage," an expert panel on programming education and development of logical thinking skills, creativity, problem-solving skills, and other abilities at the elementary school level. There, it was defined as "the ability to think logically about what sort of operations must be combined, how to combine the symbols corresponding to each operation, and how to improve the combination of symbols in order to more closely approach the intended series of actions one wants to achieve"[15].

Finally, we will survey research on transcription learning.

The original meaning of "transcription" (*shakyou* in Japanese) is to copy the Buddhist sutras (scriptures). This has a religious connotation, but here transcription refers only to the overt act of "copying symbols or characters." That is, it refers to the task of looking at sample program code, and entering it as is from the keyboard into the computer. This method involves writing a program by inputting program code and developing an understanding of the program as it executes. The method is called "transcription programming"[16]. It is also called "transcription learning" due to the fact that learning is done through the act of transcription [17]. Kita et al. created a C language programming workbook using transcription learning [18]. In this case, students enter and execute entire samples from the textbook, so they become accustomed to programming through a learning method of "becoming

accustomed rather than being taught." However, Okamoto et al. have pointed out that, with transcription learning alone, learners sometimes simply memorize the instructions and operations, and do not attain the level of understanding concepts and function. Also, instructional materials for programming learning have been developed with a focus on "visual manifestation" for conceptual understanding, and their effectiveness has been evaluated and confirmed [19]. Okamoto et al. have attempted to apply this to imitation of the problem-solving process, from the beginning stage of learning how to write code, syntax, and so on, and have achieved results with some degree of success [20]. However, Iwasaki has pointed out that effective results were not always achieved in educational practice combining video instructional materials and transcription learning [21].

### III. RESEARCH QUESTION

Okamoto et al. have noted the drawback that, when using transcription learning only, learners often just memorize the coding method for achieving a specific type of processing, and then cannot use (apply) the techniques at the stage where they create their own programs[19]. In one report supporting that view, it has been pointed out that "While students who learned with concrete examples were unable to apply that knowledge to new situations, students who learned the same concepts using abstract symbols were more often able to apply their knowledge to different situations" [22]. As one method of overcoming this issue, instructional aids (microcomputer boards) have been developed with the aim of visual manifestation, and their effectiveness has been shown. However, educational techniques employing hardware have the downside of incurring a certain degree of cost.

Thus, this study proposes "bidirectional transcription learning" as a learning method for promoting concept learning from examples. That is, in addition to transcription learning where program code is input and then executed, the aim is to achieve concept transfer by performing the inverse process of generating code from an explanation of similar code, based on the procedure of abstraction through description of code using natural language. Students go through a bidirectional procedure of converting from program code to explanations of program code, and converting from program explanations to program code, so this approach is called "bidirectional transcription learning."

The problem posed by this research is: "Does bidirectional transcription learning yield deeper understanding than transcription learning?" The purpose of this research is to examine the method's effectiveness.

### IV. METHOD

A model for bidirectional transcription learning was designed, and based on that we carried out classroom implementation, and examined effectiveness based on questionnaire results.

#### A. Model for bidirectional transcription learning

The typical procedure for transcription learning is as follows:

- 1) *Input a program provided as a sample from the keyboard into the computer.*
- 2) *Compile and execute program.*

3) Look at the results of execution, and check whether the desired results have been obtained.

4) Read the program commentary, and develop an understanding of the function and role of the program code.

Caution is necessary because a debugging situation will arise if an error occurs in step 2).

Even if the student progresses fine from step 1) to 4), there are many difficulties for beginners in step 4) like the following:

- It is hard to immediately understand the role and function of words that appear as program code. That is, even if the same word appears, it may be hard to discriminate due to the mixing of keywords and variables.
- Operations differ due to the diverse linkages between words, and there is a need for understanding adapted to the patterns of the program code.
- Program code includes, in addition to comparatively easy to recognize elements such as characters, words, and statements, concepts that aren't visible to the eye such as logical blocks and scope.

As a method for promoting concept transfer, this research proposes "bidirectional transcription" to bridge the gap between the concrete expressions of program code, and the abstract concepts contained in that code.

With bidirectional transcription, a worksheet is prepared on paper media, with a front and back. A sample program from the textbook is listed on the front of the worksheet, and an explanation of a program with similar content is provided on the back.

"Bidirectional transcription (front)" is given as the title on the front of the worksheet. The sample program code is given under that, on the left side, and to the right an empty space is provided for each line where the student can write in an explanation (Fig. 1).

Title	
Overview of sample program	
Sample program code	Space for explanation (empty space)

**Fig. 1: Worksheet form "Bidirectional transcription (front)"**

"Bidirectional transcription (back)" is given as the title on the back of the worksheet. An explanation is given under that, on the left side, for each line corresponding to the program code to be created, and to the right an empty space is provided where the student can write in program code corresponding to the explanation for each line.

Title	
Overview of program to be created	
Explanation of program to be created	Space for program code (empty space)

**Fig. 2 Worksheet form "Bidirectional transcription (back)"**

The typical procedure for bidirectional transcription learning using a worksheet is as follows:

- 1) To the side of the program listed on "Bidirectional transcription (back)", write in an explanation in natural language while referring to the textbook, etc.
- 2) To the side of the program explanation in natural language written on "Bidirectional transcription (back)" write in the program while referring to the program listed on "Bidirectional transcription (front)".
- 3) From the keyboard, input into the computer the program given on "Bidirectional transcription (front)" and the program listed on "Bidirectional transcription (back)".
- 4) Compile and execute each program.
- 5) Look at the results of execution, and check whether the desired results have been obtained.
- 6) If the desired results have not been obtained, redo from step 1) while speculating about the reason.

**B. Classroom implementation**

Classroom implementation was carried out using the proposed technique, bidirectional transcription learning, and afterwards an anonymous questionnaire was administered.

- Subjects: Second year students in the Department of Business Administration, National Institute of Technology, Ube College in academic year 2019 (number of valid responses: 45)
- Experience of learning programming among subjects: During the 1st term (April to May 2019), students learn Python programming in classes that are 90 mins. x 2 times a week x 7 weeks. They have no experience of learning programming prior to that.
- Classroom practice: 90 mins. x 4 times a week x 4 weeks in the 2nd term (June 2019)
- Overview of implementation: In the 1st term, the teaching methods combined practice problems with ordinary transcription learning. In the 2nd term, in contrast, important items in the textbook were first explained for each topic, and then the class used bidirectional transcription learning as the main approach for the applicable scope to be taught.
- Date of questionnaire administration: July 3, 2019

A scene of bidirectional transcription learning is shown in Fig. 3. Here a student is trying to write an explanation of the program listed on the "Bidirectional transcription (front)" worksheet, while referring to the textbook and other resources.



**Fig. 3: Scene of bidirectional transcription learning**

**V. RESULTS**

The results of the questionnaire on classroom implementation are indicated below.

*A. Questionnaire items*

The questionnaire items were as follows:

- Q1. Do you think the learning method of bidirectional transcription is useful for learning programming?
- Q2. If there were learning methods or materials (including textbooks) that were useful to you in the programming learning process, please indicate them together with the reason.

*B. Questionnaire results*

Questionnaire results for multiple choice responses are shown in Table 1.

The results for Q1 were each divided into positive and negative responses, and a population rate test was carried out. The result was  $P < .01$ , and it is evident that the responses were generally positive.

**TABLE I. QUESTIONNAIRE RESULTS**

	<i>Response</i>	<i>Number of responses</i>	<i>Response percentage</i>
Q1	I agree	19	42%
	I agree somewhat	18	40%
	I don't really agree	6	13%
	I don't agree	2	4%

Free responses are indicated below. Figures in parentheses are the number of respondents, including responses with the same meaning.

- Q1 response results
  - By writing an explanation based on a program, I understood the meaning of the program, and by writing a program based on an explanation, I understood the meaning of the explanation. (2)
  - I gained the ability to think on my own. (4)
  - The worksheet has a front and back, and you can work while looking at the sheet, so it's easy to understand. (2)
  - I couldn't apply my knowledge based on the textbook alone, but the process of writing enabled me to understand.
  - I gained a deeper understanding. (6)
  - By writing in my own hand, I was able to understand more readily than by just typing in. (7)
  - I gained a deeper understanding, but I feel like an electronic worksheet would be more efficient.
  - Even though I confirmed the operations, I wasn't able to understand.
- Q2 response results
  - Textbook example: Because I was able to solve the practice problem based on the example in the textbook.
  - Worksheet: It's easy to approach a friend and ask a question. It's also easy to find mistakes.
  - Worksheet: It would have been easier to write once by hand, and do an application problem on the back. (2)
  - Program execution: Because I only felt (incorrectly) that I understood, and there were many things I wasn't able to do. (2)

**VI. DISCUSSION**

Combining "I agree" and "I agree somewhat," 82% of the students had a generally positive response regarding Q1. In the free response; six students wrote that they "gained a deeper understanding." There was one student who gained deeper understanding, but questioned whether learning efficiency is good. There was also one student who responded "I couldn't understand even though I confirmed operation." However, the course of the student's learning process and the degree of understanding are unclear.

Generally speaking, it seems that bidirectional transcription learning was regarded as effective for understanding program code and creating programs at the exercise level.

In addition, there were seven students who responded that "By writing in my own hand, I was able to understand more readily than by just typing in," and this suggests applications to the "programming unplugged" efforts we are working on.

There were only free responses to question Q2, and thus the number of responses increased, but nevertheless there were responses pointing out that program execution is important in the programming learning process. This suggests the importance of debugging work, and there may be a need to examine techniques for promoting debugging work.

## VII. CONCLUSION

Classroom implementation was carried out for bidirectional transcription learning, an approach proposed independently by the authors, in an introductory course of programming education for second year students of a college of technology, who correspond to second year high school students. The results of a questionnaire showed a certain degree of positive evaluation regarding use of the proposed technique at the introductory level.

Issues for the future include adoption of the learning framework called "programming unplugged," as a technique linked to devising algorithms for solving problems, the next step in learning programming.

### Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 19K03104 .

### REFERENCES

- [1] Code.org, "President Obama asks America to learn computer science", <https://www.youtube.com/watch?v=6XvmhE1J9PY>.
- [2] Ministry of Education, Culture, Sports, Science and Technology, "Research on programming education in foreign countries", Ministry of Education, Culture, Sports, Science and Technology 2014 Information education guidance improvement support project, [http://www.mext.go.jp/a\\_menu/shotou/zyouhou/detail/\\_icsFiles/afieldfile/2018/08/10/programming\\_syogai\\_koku\\_houkokusyo.pdf](http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/_icsFiles/afieldfile/2018/08/10/programming_syogai_koku_houkokusyo.pdf). (in Japanese)
- [3] Headquarters for Japan's Economic Revitalization, "Japan Revitalization Strategy 2016", [https://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/hom\\_bun1\\_160602\\_en.pdf](https://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/hom_bun1_160602_en.pdf).
- [4] Tatsuhiro Tamaki, Makoto Tanabe, Atsushi Onishi, Makoto Sakamoto and Yasuo Uchida, "From Natural Language to Programming Language: A Stepwise Educational Method for Algorithms", *Advances in Education Research*, Volume 90, pp.9-14, 2016.
- [5] Toyohisa Nakada, "Analysis of learning computer program based on transcribing codes", *Proceedings of the Annual Conference of JSAI*, Vol.27, pp.1-3, 2013. (in Japanese)
- [6] Masako Okamoto, "Study of Computer Programming Education for Novices Focusing on Importance of Imitative Learning", *Dissertation*, pp.1-78, 2014. (in Japanese)
- [7] Essi Lahtinen, Kirsti Ala-Mutka, Hannu-Matti Järvinen, "A Study of the Difficulties of Novice Programmers", *ACM SIGCSE Bulletin* 37(3), pp.14-18, 2005.
- [8] Yayoi Hofuku, "'Peta-gogy" for Future : Why is Programming Difficult?", *IPSJ Magazine*, Vol.54, No.3, pp.252-255, 2013. (in Japanese)
- [9] Shinya Cho, Yayoi Hofuku, Tomohiro Nishida, Susumu Kanemune, "De-gapper — Tool for Support Programming Learners' "Step-by-step" Learning", *IPSJ Journal*, Vol.55, No.1, pp.45-56, 2014. (in Japanese)
- [10] Koji Komatsu, "Problems and Measures of Programming Education", *Bunkyo Gakuin University Research Institute Management Review*, Vol.25, No.1, pp.83-104, 2015. (in Japanese)
- [11] Matsubara Yasuo, "Cognitive Scientific Study of Programming (1)", *Information and communication studies*, Vol.7, pp.96-104, 1986. (in Japanese)
- [12] Go Ota, Yosuke Morimoto, Hiroshi Kato, "The Comparative Survey of Computer Science and Programming Education for Primary and Secondary Schools in the UK, Australia and USA", *Japan Journal of Educational Technology*, Vol.40, No.3, pp.197-208, 2016. (in Japanese)
- [13] Jeannette M. Wing, "Computational Thinking", *IPSJ Magazine*, Vol.56, No.6, pp.584-587, 2015. (in Japanese)
- [14] Kotatsu Rin, "A Review of the State of Computational Thinking Discourse", *Research report of JET Conferences*, Vol.18, No.2, pp.165-172, 2018. (in Japanese)
- [15] Ministry of Education, Culture, Sports, Science and Technology, "About the Way of Programming Education in the Elementary School Stage", [http://www.mext.go.jp/b\\_menu/shingi/chukyo/chukyo\\_3/074/siryu/\\_icsFiles/afieldfile/2016/07/07/1373891\\_5\\_1\\_1.pdf](http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo_3/074/siryu/_icsFiles/afieldfile/2016/07/07/1373891_5_1_1.pdf), 2016. (in Japanese)
- [16] Toyohisa Nakada, "Analysis of learning computer program based on transcribing codes", *The 27th Annual Conference of the Japanese Society for Artificial Intelligence*, 203-2in, pp.1-3, 2013. (in Japanese)
- [17] Masako Okamoto, "Study of Computer Programming Education for Novices Focusing on Importance of Imitative Learning", *Dissertation*, Kyoto University, pp.1-80, 2014. (in Japanese)
- [18] Hajime Kita, Masako Okamoto, Takeshi Fujioka, Naoto Yoshikawa, "C language programming workbook by copying and learning", *Kyoritsu Shuppan*, 2012. (in Japanese)
- [19] Masako Okamoto, Masayuki Murakami, Naoto Yoshikawa, Hajime Kita, "Development and Assessment of Learning Materials for Computer Programming Focusing on the "Visual Manifestation"", *Japan Journal of Educational Technology* 37(1), 35-45, 2013. (in Japanese)
- [20] Masako Okamoto, Takeshi Fujioka, Hajime Kita, "Development of Teaching Material Using Agent-Based Simulation for Problem-Solving based Informatics in Senior-High School", *Japan Journal of Educational Technology*, Vol.35, Issue Suppl, pp.97-100, 2011. (in Japanese)
- [21] Hideo Iwasaki, "Video Teaching Materials and Shakyo-Style Learning for Computer Programming Courses—A Study on Teaching Using Digital Textbooks in Higher Education(2)—", *J. Higher Education*, Tokai University (Hokkaido Campus) 17, 2017. (in Japanese)
- [22] Kaminski Jennifer A., Vladimir M. Sloutsky, and Andrew F. Heckler, "The Advantage of Abstract Examples in Learning math", *Science*, No.320, pp.454-455, 2008.