International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

INTERNATIONAL JOURNAL OF MODERN EDUCATION (IJMOE)
www.ijmoe.com

GLOBAL ACADEMIC EXCELLENCE

LEARNERS' CONFESSION FOR BIDIRECTIONAL TRANSCRIPTION EFFECTIVENESS FOR BEGINNERS IN THE PROGRAMMING COURSE

Tatsuhiro Tamaki[1], Harumi Hashimoto[2], Atsushi Onishi[3], Yasuo Uchida[4*]

[1] Department of Media Information Engineering, National Institute of Technology, Okinawa College, Japan
Email: t.tamaki@okinawa-ct.ac.jp
[2] Department of Business Administration and Information, Setsunan University, Japan
Email: harumi.hashimoto@kjo.setsunan.ac.jp
[3] Department of Integrated Science and Technology, National Institute of Technology, Tsuyama College, Japan
Email: a-onishi@tsuyama-ct.ac.jp
[4] Department of Business Administration, Miyazaki Sangyo-keiei University, Japan
Email: uchida@mail.miyasankei-u.ac.jp
[*] Corresponding Author

Article Info:

Article history:
Received date: 15.07.2021
Revised date: 25.07.2021
Accepted date: 03.08.2021
Published date: 05.09.2021

To cite this document:

Tamaki, T., Hashimoto, H., Onishi, A., & Uchida, Y. (2021). Learners' Confession For Bidirectional Transcription Effectiveness For Beginners In The Programming Course. *International Journal of Modern Education, 3*(10), 32-47.

DOI: 10.35631/IJMOE.310003

Abstract:

The 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 is 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. It focuses on the process of converting a natural language to a programming language in the final stage of unplugged to code writing. 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, Unplugged

International Journal of Modern Education
IJMOE
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Introduction

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 (Code.org, 2013), and steps such as making programming education mandatory from the compulsory education stage have already begun in the U.K. and other countries (Ministry of Education, Culture, Sports, Science and Technology, 2018).

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 (Headquarters for Japan's Economic Revitalization, 2016). 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 (Tamaki, Tanabe, Onishi, Sakamoto, & Uchida, 2016). First is the process of devising algorithm. 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 between 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 (Nakada, 2013; Okamoto, 2014), but we have conducted a trial to further improve efficiency and deepen understanding, and here we provide an overview and report on our results.

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 (Lahtinen, Ala-Mutka, & Järvinen, 2005). There is also research posing and discussing the question "Why is programming difficult?" (Hofuku, 2013). 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 (Cho, Hofuku, Nishida, & Kanemune, 2014). It is also shown to be effective to analyse 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 (Komatsu, 2015). The problems with conventional programming

International Journal of Modern Education
IJMOE
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

education are identified and analysed, 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 (Matsubara, 1986). 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 (Ota, Morimoto, & Kato, 2016). 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, 2015). 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) (Rin, 2018).

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" (Ministry of Education, Culture, Sports, Science and Technology, 2016).

International Journal of Modern Education
IJMOE
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

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" (Nakada, 2013). It is also called "transcription learning" due to the fact that learning is done through the act of transcription (Okamoto, 2014). Kita et al. created a C language programming workbook using transcription learning (Kita, Okamoto, Fujioka, & Yoshikawa, 2012). 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, Murakami, Yoshikawa, & Kita (2013) 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. Okamoto, Fujioka, & Kita, (2011) 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 . However, Iwasaki has pointed out that effective results were not always achieved in educational practice combining video instructional materials and transcription learning (Iwasaki, 2017).

Relative Work

The 6-step method we are proposing, which is the basis of this research, will be explained (Tamaki, Onishi, & Uchida, 2021). The 6-step method is a method for expanding from CS Unplugged to full- fledged programming. That is, Step1: CS Unplugged → Step2: CS Plugged → Step3: Illustration of Activity processing process (visualization) → Step4: Natural language description of Activity processing process (abstraction, element extraction) → Step5: Added original expression It is a teaching method of confirming the operation of the algorithm by table tracing (verification) → Step 6: Writing full- fledged program code (abstraction, coding).

The CS Unplugged assigned to Step 1 is said to be effective for information science education. CS Unplugged is a teaching method for teaching information science without using a computer, advocated by Tim Bell et al. of the University of Canterbury, New Zealand (Bell, Witten, Fellows, Adams, & McKenzie, 2005). A study by Tim Bell et al. created a guidebook for applying CS Unplugged to elementary school children. Subsequent studies can be categorized into several approaches. That is, (1) Research that analyzes and researches the educational method itself called CS Unplugged (including those that generalize Unplugged), (2) Research that practices CS Unplugged and aims at its educational effect, (3) Research that devises a new activity of CS Unplugged , and (4) Research that CS Plugged Tools, (5) Research that combines CS Unplugged and other educational methods, (6) Research that aims to develop from CS Unplugged to full- fledged programming (the subject of this research). Related studies include the use of teaching tools in CS Unplugged algorithm learning (Manabe, Kanemune, & Namiki, 2013). There is a study as a practice for high school students, and a limited but successful report has been made (Feastery, Segarsz, Wahbay, & Hallstrom, 2011). However, most of the

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

conventional research ends in the experience stage of CS Unplugged or is limited from the viewpoint of connectivity to full-fledged programming education.

Figure 1 shows "Activity1: Counting Numbers (Binary Number)" as a practical example of "Step1: CS Unplugged", which is the start of the 6-step method.

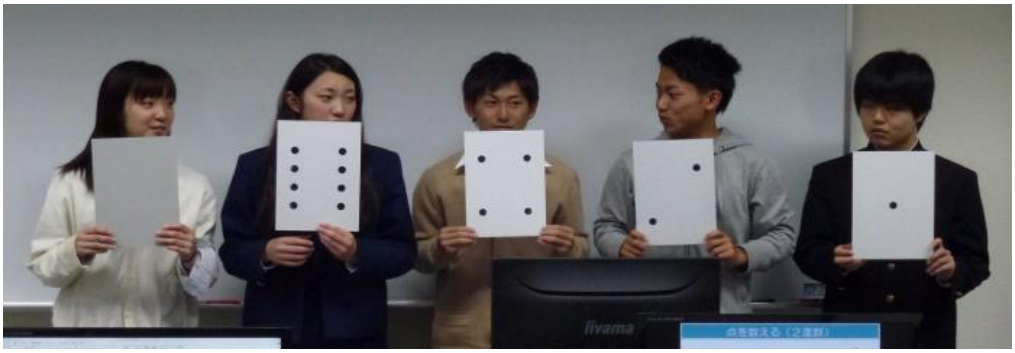

Figure 1: Practice of "Counting Numbers (Binary Number)"

As a term we define independently, we will call the realization of CS Unplugged Activity using a computer "CS Plugged ". "CS Plugged " is the main method for "acquiring an image" for processing, which is a prerequisite for programming. We propose a method called "CS Plugged " as a new approach to complement CS Unplugged. In other words, the basic idea of CS Unplugged is "do not use a computer", but in order to expand to full-fledged programming, we aim to supplement it with a "computer-based" educational method that acts as a bridge.

As a prototype example of "Step2: CS Plugged tool", Figure 2 is shown in which "Activity2: representing a colour by a number (image representation)" is implemented.

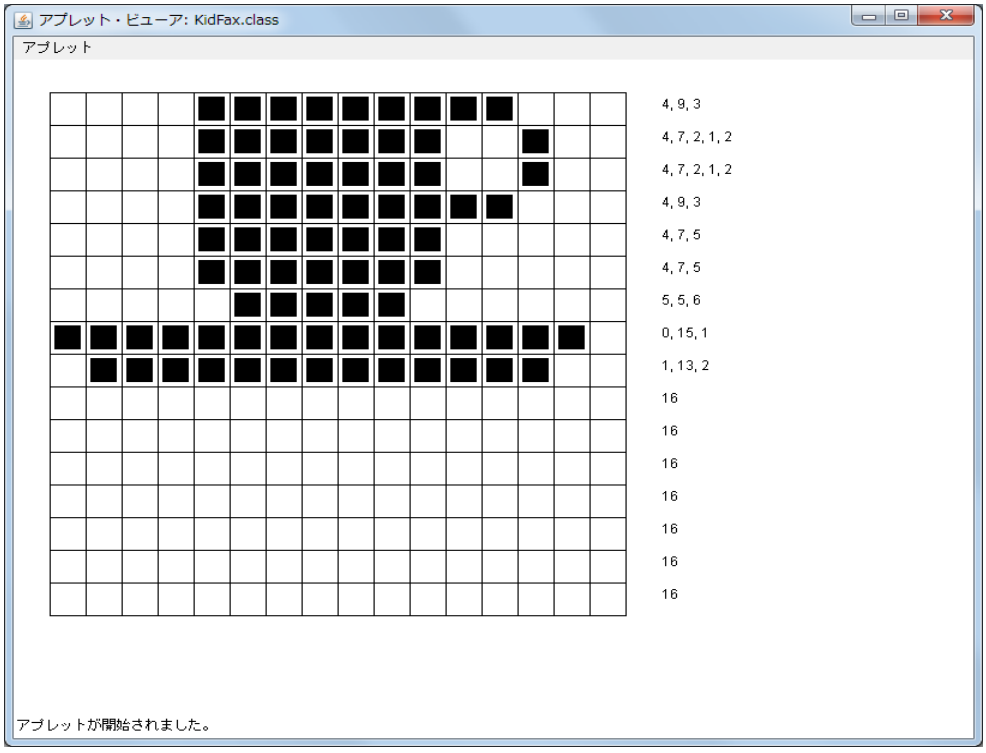

Figure 2: Prototype "CS Plugged Tool"

International Journal of Modern Education
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Figure 3 shows an illustration of "Finding the Sum from 1 to n" as a drawing example of "Step 3: Illustration (Visualization) of the processing process of Activity".

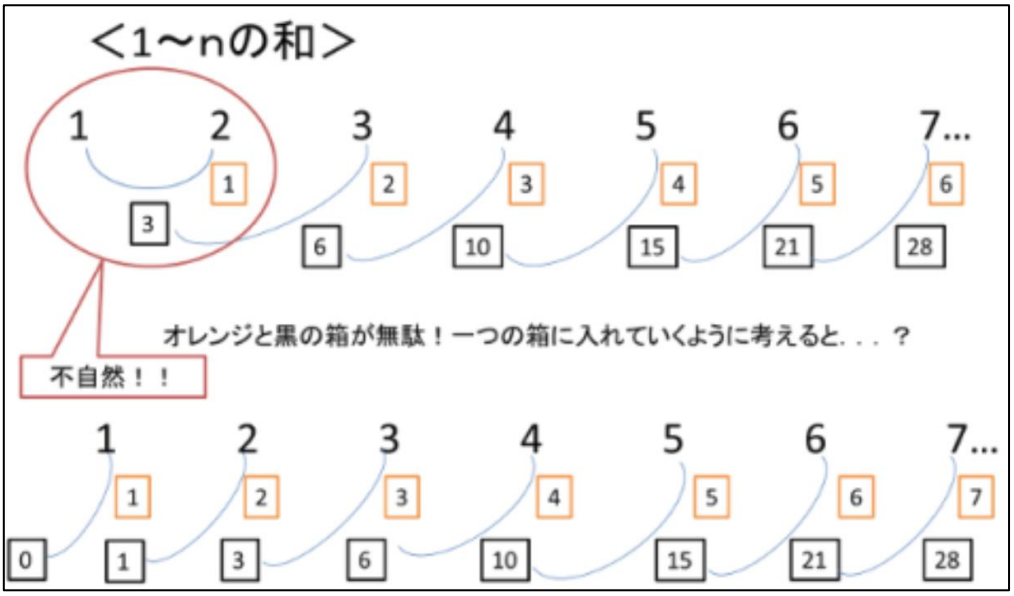

Figure 3: "Finding the Sum from 1 to n" Illustrated

Figure 4 shows an example of describing the process of "Finding the sum from 1 to n" in natural language as a description example of "Step 4: Natural language description (abstraction, element extraction) of the processing process of activity".

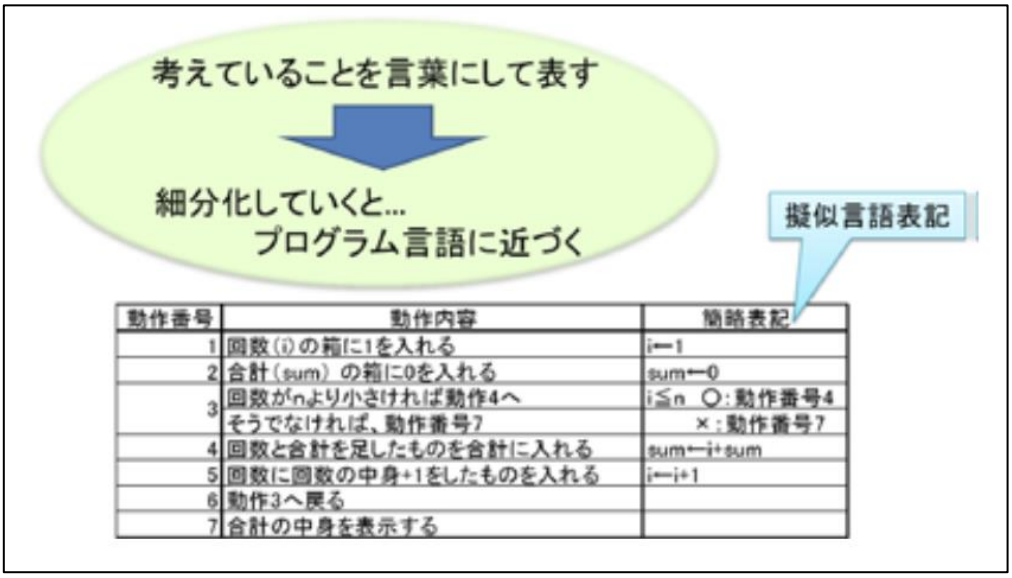

Figure 4: Natural Language Description of the Process of "Finding the Sum from 1 to n"

Figure 5 shows an example of creating "Activity2: Representing colours by numbers (image representation)" as "Step5: Confirming the operation of the algorithm by table tracing with a unique expression (verification)".

International Journal of Modern Education
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Array

	0	1	2	3	4
0	□	■	■	■	□
1	□	□	□	□	■
2	□	■	■	■	■
3	■	□	□	□	■
4	■	□	□	□	■
5	□	■	■	■	■

Subscript i	Subscript j	Array[i][j]	Previous character	Count
0	0	□	□	1
0	1	■	□	1
0	2	■	■	2
0	3	■	■	3
0	4	□	■	1
1	0	□	□	1
1	1			
1	2			
1	3			
1	4			
…	…			
5	0			
5	1			
5	2			
5	3			
5	4			

Output

□	■	■	■	□	1,3,1
□	□	□	□	■	

Figure 5: Sample Trace Table (in process)

Finally, as an example of "Step 6: Full-fledged program code description (abstraction, coding)", Figure 6 shows an example of converting the process of "Finding the sum from 1 to n" into the Java Language.

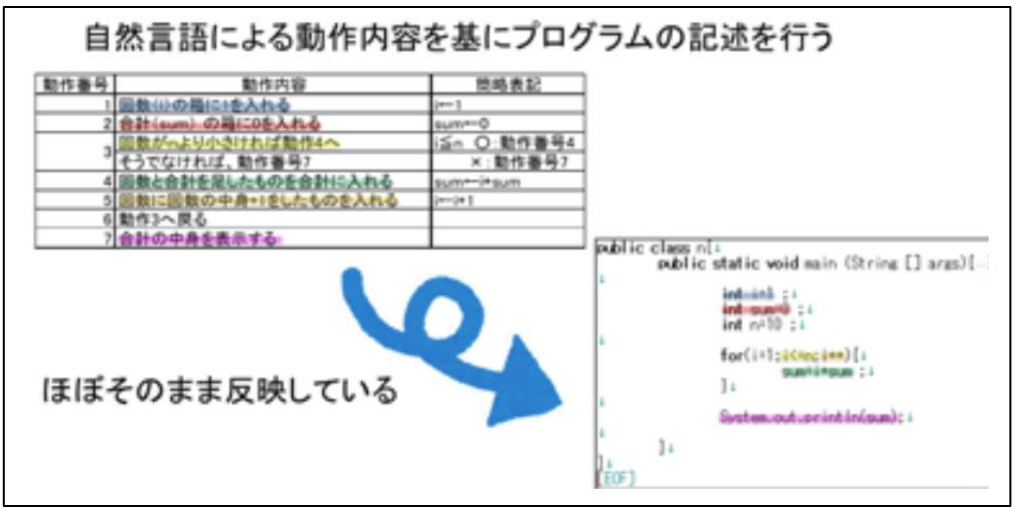

Figure 6: Conversion of the Natural Language Description of the "Find the Sum from 1 to n" Process to the Java Language

Research Question

Okamoto et al. (2013) 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.

International Journal of Modern Education
IJMOE
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

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" (Kaminski, Vladimir, & Andrew, 2008). 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.

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.

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.

International Journal of Modern Education
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

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 (Figure 7).

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

Figure 7: 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 (Figure 8).

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

Figure 8: 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 (front)", 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 "Bidirection transcription (front)" and the program listed on "Bidirectional transcription (back)".
4) Compile and execute each program.

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

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.

Curriculum Overview

Table 1 shows the outline of the curriculum in this practice, which incorporates bidirectional transcription learning as a method that complements "Step 6: Full-fledged program code description (abstraction, coding)" among the 6-step methods. The scope of this time is for the initial stage of programming learning.

Table 1: Outline of Curriculum Incorporating Bidirectional Transcription Learning

Times	Basic Elements	Contents
1	Iterative	For loop, Display of list elements
2	Iterative	For loop, Range object
3	Iterative	While loop, Double loop
4	Lists and tuples	Double loops and 2D arrays (List of lists)
5	Utilization of loops	Break, Continue
6	Utilization of loops	Enumerate () function, Zip () function
7	Exception handling	Try ～ except

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 Figure 9. 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.

International Journal of Modern Education
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

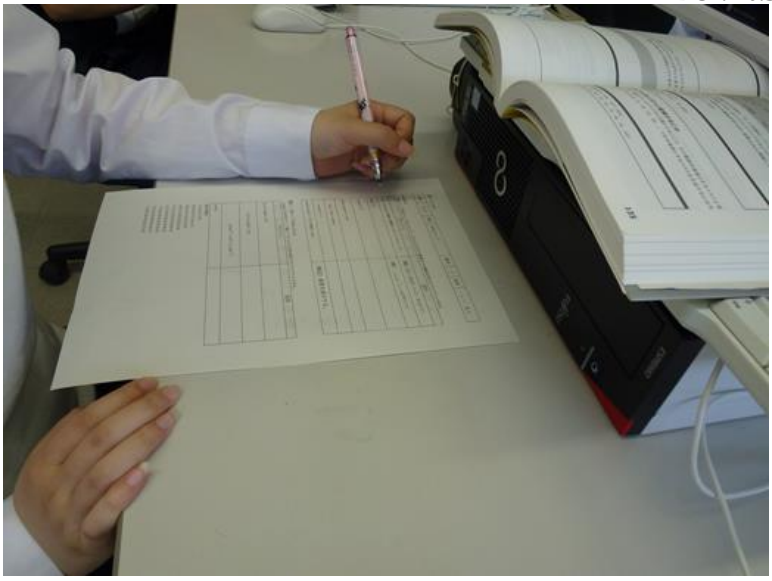

Figure 9: Scene of Bidirectional Transcription Learning

Results

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

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.

Questionnaire Results

Questionnaire results for multiple choice responses are shown in Figure 10.

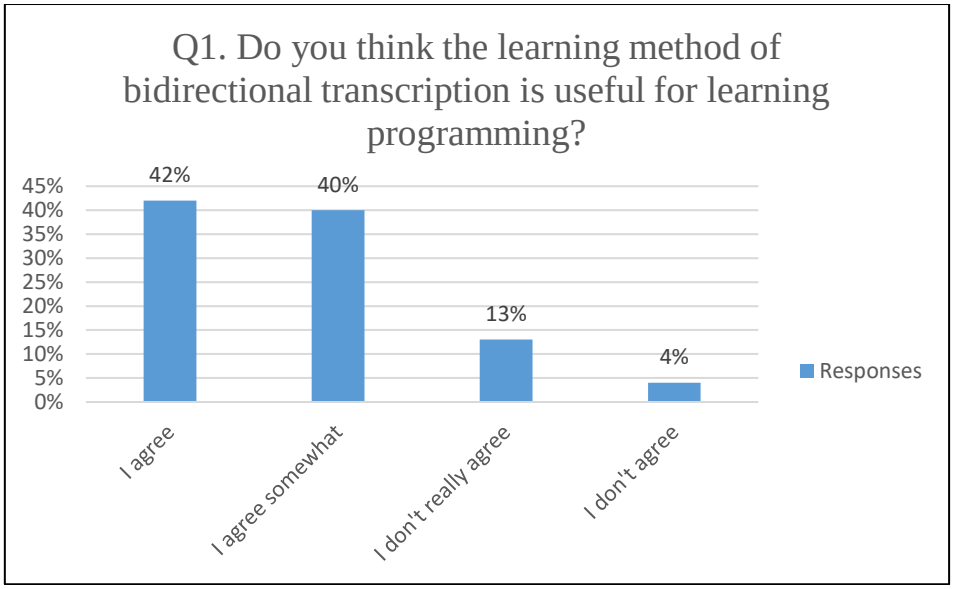

Figure 10: Questionnaire Results for Q1

International Journal of Modern Education
IJMOE
EISSN: 2637-0905
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

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.

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 I 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 that 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)

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 that by just typing in," and this suggests applications to the "programming unplugged" efforts we are working on.

Watanabe & Takemura (2019) discusses the relationship between natural and programming languages. In particular, the foundation of human speculative activity is human language

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

(natural language), and the speculative process occurring in the specialized technical field of programming is attracted to the foundation of human language (natural language) and re-created. I'm trying to build. He points out that the program can be executed by considering the source code as a program, with the expression "deemed". Indeed, the act of " bidirectional transcription" in this research is in line with this, and the source code, which is essentially just text data, is converted into data as a program that is executed as written there. It is understood that it is converted to. From free responses of the questionnaire results, it can be read that it may contribute to the promotion of deeper abstraction formation compared to unidirectional copying.

The learning steps of the 6-step method start from unplugged (Tamaki, Onishi, & Uchida, 2021). In other words, after the problem to be solved is defined, it is the stage to determine the policy of how to solve it. At this time, it is the role of unplugged to actually follow the path of human problem solving. At this stage, the "behavior" of the processing process can be recognized, but the expression in natural language has not yet been materialized. The final sixth step is the process of converting natural language to source code, but it is not just a translation, but a process that involves the "behavior" of the behavior. Even at this stage, it can be inferred that the inclusion of the unplugged element of bidirectional conversion between natural language and source code led to a deeper understanding of programming.

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.

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. In particular, it can be said that it helped to form a thinking process that converts the abstract concept of instructions and instructions to the computer behind the natural language description as source code.

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

Bell, T. Witten, I.H. Fellows, M. Adams, & R. McKenzie, J. (2005). Computer Science Unplugged: An enrichment and extension programme for primary-aged children. Retrieved from https://ir.canterbury.ac.nz/handle/10092/247.

Cho, S., Hofuku, Y., Nishida, T., & Kanemune, S. (2014). De-gapper—Tool for Support Programming Learners' “Step-by-step” Learning. *IPSJ Journal*, 55(1), 45-56.

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Code.org. (2013). President Obama asks America to learn computer science. Retrieved from https://www.youtube.com/watch?v=6XvmhE1J9PY

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

Headquarters for Japan's Economic Revitalization. (2016). Japan Revitalization Strategy 2016. Retrieved from https://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/hombun1_160602_en.pdf

Hofuku, Y. (2013). "Peta-gogy" for Future: Why is Programming Difficult?. *IPSJ Magazine*, 54(3), 252-255.

Iwasaki, H. (2017). 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.

Kaminski, J. A., Vladimir, M. S., & Andrew, F. H. (2008). The Advantage of Abstract Examples in Learning math. *Science*. *3*20, 454-455.

Kita, H., Okamoto, M., Fujioka, T., & Yoshikawa, N. (2012). *C language programming workbook by copying and learning*. Tokyo, Japan: Kyoritsu Shuppan.

Komatsu, K. (2015). Problems and Measures of Programming Education. *Bunkyo Gakuin University Research Institute Management Review*, 25(1), 83-104.

Lahtinen, E., Ala-Mutka, K., & Järvinen H-M. (2005). A Study of the Difficulties of Novice Programmers. *ACM SIGCSE Bulletin*, 37(3), 14-18.

Manabe, H., Kanemune, S., & Namiki, M. (2013). Effects of Teaching Tools in CSU Algorithm Education, *IPSJ Journal*, 54(1), 14-23.

Matsubara, Y. (1986). Cognitive Scientific Study of Programming (1). *Information and communication studies*, 7, 96-104.

Ministry of Education, Culture, Sports, Science and Technology. (2016). About the Way of Programming Education in the Elementary School Stage. Retrieved from http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo3/074/siryo/__icsFiles/afieldfile/2016/07/07/1373891_5_1_1.pdf

Ministry of Education, Culture, Sports, Science and Technology. (2018). Research on programming education in foreign countries. Ministry of Education, Culture, Sports, Science and Technology 2014 Information education guidance improvement support project. Retrieved from http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/__icsFiles/afieldfile/2018/08/10/programming_syogaikoku_houkokusyo.pdf

Nakada, T. (2013). Analysis of learning computer program based on transcribing codes. *Proceedings of the Annual Conference of JSAI*. 27,1-3.

Okamoto, M., Fujioka, T., & Kita, H. (2011). Development of Teaching Material Using Agent-Based Simulation for Problem-Solving based Informatics in Senior-High School. *Japan Journal of Educational Technology*. 35(S), 97-100.

Okamoto, M., Murakami, M., Yoshikawa, N., & Kita, H. (2013). Development and Assessment of Learning Materials for Computer Programming Focusing on the "Visual Manifestation". *Japan Journal of Educational Technology*. 37(1), 35-45.

Okamoto, M. (2014). Study of Computer Programming Education for Novices Focusing on Importance of Imitative Learning. *Dissertation*. Kyoto University, 1-80.

Ota, G., Morimoto, Y., & Kato, H. (2016). 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*. 40(3), 197-208.

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Rin, K. (2018). A Review of the State of Computational Thinking Discourse. *Research report of JET Conferences*. 18(2), 165-172.

Tamaki, T., Tanabe, M., Onishi, A., Sakamoto, M., & Uchida, Y. (2016). From Natural Language to Programming Language: A Stepwise Educational Method for Algorithms. *Advances in Education Research*. 90, 9-14.

Tamaki, T., Onishi, A., & Uchida, Y. (2021). A Trial of Learning Programming Using a Six-step Method, *Asian Journal of Research in Education Social Sciences*, 3(1), 10-24.

Watanabe, H. & Takemura, T. (2019). As were it natural language : How all is presumed in programming languages, Hitotsubashi review of arts and sciences ,(13), 146-189

Wing, J. M. (2015). Computational Thinking. *IPSJ Magazine*, 56(6), 584-587.

International Journal of Modern Education
EISSN: 2637-0905
IJMOE
Volume 3 Issue 10 (September 2021) PP. 32-47
DOI: 10.35631/IJMOE.310003

Appendix 1: Questionnaire of "Q1. Do you think the learning method of bidirectional transcription is useful for learning programming?"

Please circle the item that best describes or reflect you based on the following statements; I agree, I agree somewhat, I don't really agree, I don't agree.

In addition, if possible, explain why you chose that option.

Questionnaire Results

	Response	Number of Responses	Response Percentage
Q1	I agree	19	42%
	I agree somewhat	18	40%
	I don't really agree	6	13%
	I don't agree	2	4%