Title	小学校段階におけるプログラミング的思考を操作として展開・評価する学習環境の構築
Author(s)	小林, 祐太
Citation	
Issue Date	2021-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/17097
Rights	
Description	Supervisor: 長谷川 忍, 先端科学技術研究科, 修士（情報科学）

The System Construction of Learning Environment for Operating Evaluation about Computational Thinking for Elementary School Students in Japan

1910095 Yuta Kobayashi

In Japan, programming education has become compulsory in primary education from 2020. The background of this effort is that the government promotes the growth of "programming thinking" acquired in the process of learning programming. Programming thinking is "the ability to think logically in order to realize a series of activities intended by oneself" and is expected as a problem-solving ability in an unpredictable society in the future.

However, one of the problems is that the evaluation method of programming thinking is not clear. This is because programming thinking is implicit, and it is difficult to observe its level and growth from the outside. Therefore, discussions have continued to this day regarding the development goals and evaluation criteria for programming thinking. In fact, Koizumi organized programming thinking into six components based on computational thinking practiced in the United Kingdom in order to formulate evaluation criteria for programming thinking. Furthermore, Benesse positioned the six elements proposed by Koizumi as learning goals for programming education.

In the conventional evaluation method, since the invisible concept of programming thinking is captured, the activities based on the programming thinking mentioned above are evaluated according to the achievement level set by the researcher. For example, Nishino created and operated a rubric that performs an objective evaluation in three stages in order to capture thoughts during programming learning based on Koizumi's evaluation criteria. Saito proposed the rubric "Rubric Pro EEs" based on a four-grade evaluation as an evaluation index for the entire programming education. However, in each case, the position is limited to the standard of achievement, and the level of programming thinking has not been quantitatively evaluated.

On the other hand, there is also a problem with training in programming education. According to the "Results of Survey on Efforts for Elementary School Programming Education by the Municipal Board of Education" published by the government in January 2020, the training of programming education for teachers nationwide was not wholly conducted. The cause is that the teaching methods and teaching materials are not sufficient. The programming education introduced in elementary school does not focus on acquiring programming abilities using programming languages such as the so-called

C language but seeks to develop logical thinking in various subjects. Since it is clearly distinguished from programming ability, it is difficult to apply the technical education provided in high school and above as it is. In addition, although there are places where programming education is researched in primary schools, it is difficult to collect information because the number is limited.

The purpose of this study is to build a learning environment in which the ability of programming thinking is evaluated by "formative evaluation" in order to grasp the degree of programming thinking during learning of elementary school children. Formative evaluation evaluates the learner's current achievement level and is one of the policies to switch the learning method depending on the evaluation. Based on this formative assessment, it is necessary to extract the process of thinking in order to grasp the degree of programming thinking. Therefore, we externalized thinking as an operation and tried to quantify the content of the operation during the exercise. Specifically, we have developed a system for extracting activities based on thoughts. As a teaching material that can be used even by teachers who do not have a clear understanding of programming education and learning methods, we aimed to create an environment where learners can surely demonstrate programming thinking by constructing the actual programming learning process in the system.

This research aims to capture the thinking when constructing the control structure in programming and capture the activities based on the elements of programming thinking that appear every time the problem is solved throughout the programming learning. It deals with learning from the design stage to looking back on the coding results and records the operations performed during the learning during that time.

In order to develop, we thought about how to acquire operations from learning based on programming thinking.

First, in order to capture the thoughts that occur during programming learning in detail, we defined six "thinking activities" that are the operations of the objects to be observed from Benesse's evaluation criteria. The defined thinking activities are "division", "abstraction", "generalization", "ordering", and "analysis".

Next, we set the learning phase assuming that these activities occur in actual programming learning. The learning phase was created in three stages: "design," "development," and "evaluation," with reference to the learning process that incorporates the concept of software design.

Finally, we designed a learning task called "thinking task" with four patterns to acquire the operation. Thinking tasks are "inference tasks" that capture abstraction and ordering, "division tasks" that capture division, "control tasks" that capture control, and "analysis tasks" that capture generalization and analysis. Then, the number of simple operations in the thinking task was defined as "quantity data," and the data calculated from the answer results were defined as "quality data," which was positioned as an index of the ability of programming thinking in Thinkron.

Based on these definitions and designs, we created a web application called "Thinkron" that can acquire learner operation logs. It is based on JavaScript and has a UI that only clicks and drags and drops the mouse to use it easily. The task execution screen is a procedural maze search game in which the procedure is easily reflected as a result.

In order to evaluate Thinkron, we used "Rubric ProEEs" created for programming education and experimented to see if there was a correlation between Thinkron's score and rubric evaluation.

As a result, a correlation was found in all thinking activities except "control" of thinking activities. This suggests that thinking activities with correlation are effective as an index for estimating the ability of programming thinking. No significant difference was observed only in the "control" of thinking activity because it was not possible to capture deductive thinking such as conditional branching during coding. Therefore, it will be necessary to consider how to extract the target code and how to score it in the future.

In addition, the effect of improving the division ability was recognized through learning using Thinkron. There are few examples of learning materials that are conscious of the design side, and the learning effect of Thinkron in this design phase will be positioned as one of the learning examples that capture the design stage that is effective for programming education. After this experiment, we also conducted a sensitivity evaluation questionnaire about the usability of Thinkron. As a result of investigating the fun of Thinkron, the difficulty of asking questions, and the difficulty of operation, it was found that the operability was easy for children to accept.