I. J. Education and Management Engineering, **2025, 5, 1-12**
Published Online on October 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijeme.2025.05.01

Assessing Algorithmic Literacy through Scratch Programming: A Comparative Study in Japanese and Indonesian STEM Education

Hiroyuki Endo*
Department of Human Education, Naruto University of Education, Tokushima, Japan
Email: endo.hiroyuki88@gmail.com
ORCID iD: https://orcid.org/0009-0000-3362-0394
*Corresponding Author

Satoshi Kusaka
Department of Human Education, Naruto University of Education, Tokushima, Japan
Email: kusaka@ksm.world
ORCID iD: https://orcid.org/0000-0002-9047-9312

Rully Charitas Indra Prahmana
Mathematics Education Department, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
Ethno-Realistic Mathematics Education Research Center, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
Email: rully.indra@mpmat.uad.ac.id
ORCID iD: https://orcid.org/0000-0002-9406-689X

Nur Robiah Nofikusumawati Peni
Mathematics Education Department, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
Ethno-Realistic Mathematics Education Research Center, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
Email: nur.peni@mpmat.uad.ac.id
ORCID iD: https://orcid.org/0000-0001-7841-6135

Received: 09 April, 2025; Revised: 28 June, 2025; Accepted: 17 September, 2025; Published: 08 October, 2025

Abstract: Despite the growing integration of programming education within STEM curricula worldwide, there remains a significant gap in the availability of comprehensive and empirically validated assessment tools, particularly those capable of evaluating students' broader computational thinking competencies. Addressing this gap, this study introduces a novel rubric grounded in the framework of algorithmic literacy—defined by Kalaš (2011) as a set of skills that enable individuals to manage behavior and attain goals within dynamic, interactive environments. The rubric, comprising four key dimensions—Subject Knowledge, Programming Design, Programming Techniques, and the Learning Process—was developed to assess students' algorithmic literacy in the context of Scratch-based STEM learning activities. The study was guided by two primary research questions: (1) To what extent can the developed rubric effectively assess students' algorithmic literacy in Scratch-based STEM education? and (2) What similarities and differences exist between Japanese and Indonesian students in their programming outcomes, and what contextual factors contribute to these patterns? A total of 228 students—132 from Japan and 96 from Indonesia—participated in three intervention lessons focused on programming regular polygons using Scratch. Evaluation of their work revealed high proficiency in Programming Techniques, particularly in utilizing sequential commands and loops, while students exhibited difficulties in integrating complex mathematical concepts and multi-shape design tasks. Comparative analysis highlighted the influence of curriculum structure, access to technological resources, and pedagogical approaches on student performance across the two contexts. The findings contribute a validated and practical tool for assessing algorithmic literacy, offering critical insights to inform the design of culturally responsive and pedagogically effective programming education initiatives.

Index Terms: Algorithmic Literacy, Indonesia, Japan, Rubric Assessment, Scratch, STEM Education

1. Introduction

Programming education was introduced in Japanese primary schools in 2020. According to the Course of Study, the Japanese curriculum outlines three main objectives for programming education: (1) To improve students' programming thinking , (2) To understand the system and advantage of program and our society is supported by Information Technology and computer, (3) To cultivate the attitude to create better society by solving problems with computer [1]. In Japan, programming education is not established as a separate subject; rather, it is incorporated into various subjects with the aim of fostering students' programing thinking. The Ministry of Education, Culture, Sports, Science, and Technology (MEXT) defines programming thinking as a concept based on computational thinking, encompassing logical reasoning required for programming. The adoption of digital devices in Japanese schools accelerated following the COVID-19 pandemic, and currently, all elementary and junior high school students have access to personal ICT devices. Consequently, educators must optimize the use of these tools in their teaching.

Globally, programming education is gaining significance, particularly in Southeast Asia, where the digital economy is projected to reach $300 billion by 2030, making it the fourth-largest economic bloc worldwide. Indonesia, the fourth most populous country, has seen major funding rounds for technology firms such as Bukalapak, Gojek, Tokopedia, and Traveloka [2]. Despite this, Indonesia has yet to formally incorporate programming education into its national curriculum. While several Indonesian schools have independently implemented computational thinking and IT literacy initiatives, these efforts remain at a nascent stage. Additionally, while STEM education is well-established in developed nations, Indonesia's government is only beginning to integrate STEM into its curriculum, and standardized assessments for STEM learning are lacking [3]. According to Hardika [4], Indonesia still is one of the countries that has less development of ICT in education. Information and Communication Technology (ICT) is still a compulsory subject for junior and senior high schools, as well as a local subject for elementary schools. The use of ICT in education is needed for Indonesia to catch up with other nations and accelerate its economic growth and competitiveness [4].

Despite the global expansion of programming education, it remains an evolving field. Japan is advancing ICT integration in classrooms, whereas Indonesia is in the early stages of STEM adoption. These differences necessitate an investigation into how programming education impacts students' algorithmic thinking development across these two countries.

Furthermore, computational thinking (CT) research is still in its early stages, and scholars have yet to reach a consensus on its precise definition, instructional methods, and assessment approaches [5]. Empirical research on evaluating students' CT through programming remains limited [6]. To support the growth of STEM education, it is crucial to not only develop teaching methodologies but also establish comprehensive assessment frameworks for programming education.

2. Literature Review

2.1 Relationship between algorithmic thinking and computational thinking

Algorithmic Thinking is a fundamental component of Computational Thinking [7]. Several researchers, including Wing [8] and Mandoye [9], have defined Algorithmic Thinking. According to Mandoye [9], Algorithmic Thinking refers to the ability to develop step-by-step procedures that can be implemented in a programming language to complete a task. He refers to this as "algorithm design." Figure 1 illustrates the image of using algorithmic thinking, developed by the author based on the theory of the programming education in primary school in Japan [7].

Children first identify a problem and set a learning task. They then apply Algorithmic Thinking to solve problems. In doing so, they utilize previously acquired knowledge and skills. Through trial and error, children solve problems by decomposing, generalizing, abstracting, and combining elements into algorithms. The Ministry of Education, Culture, Sports, Science, and Technology (MEXT) uses the term "programming thinking," stating that "programming thinking develops progressively through repeated learning"[7]. In other words, within the four components of Computational Thinking, Algorithmic Thinking can be regarded as the ability to comprehensively integrate the other three components: decomposition, generalization, and abstraction.

With the integration of Algorithmic Thinking into education, coding training has become an essential part of curricula. The rapid advancement of science and technology underscores the increasing necessity of coding and sequential processing. Therefore, fostering Algorithmic Thinking in both students and teachers is crucial for the future of education and society [10].

Algorithmic Thinking is an increasingly important cognitive skill in the modern world. Beyond computer-related tasks, people encounter challenges in interpersonal interactions, local communities, and global issues. Regardless of the context, Algorithmic Thinking provides a structured approach to identifying clues and directions for solutions. While it is a key competency for leveraging artificial intelligence, its applicability extends far beyond that—it is a critical skill for addressing a wide range of real-world problems.

For these reasons, this study focuses on Algorithmic Thinking as a fundamental aspect of Computational Thinking.

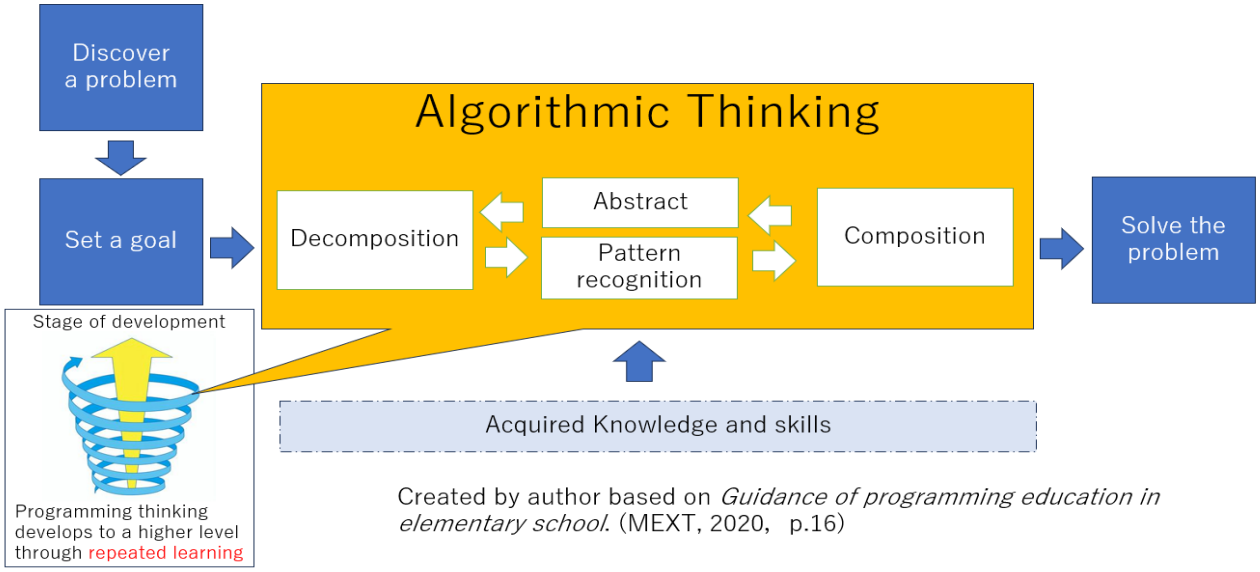


Fig. 1. Image of using Algorithmic Thinking

2.2 Algorithmic Thinking in Scratch

Among the four components of Computational Thinking, Algorithmic Thinking can be seen as the ability to effectively integrate decomposition, generalization, and abstraction. A practical example from daily life is cooking curry and rice, which involves multiple steps such as cutting, stir-frying, and serving vegetables. The process of "making curry rice" can be broken down into specific tasks: for instance, "cutting vegetables" includes "cutting carrots," "cutting potatoes," and "cutting onions." Recognizing that these distinct steps share a common principle—"cutting"—demonstrates generalization. The ability to identify the overarching principle behind these actions exemplifies abstraction. Decomposing, generalizing, and abstracting a sequence of tasks and systematically combining them form the foundation of an algorithm.

Algorithmic Thinking is not only applicable in daily life scenarios such as cooking and giving directions but also plays a crucial role in academic subjects, including mathematics. For example, solving the division problem $72 \div 3$ involves a sequence of steps: identifying groups of 2, multiplying 3 by 2, subtracting 6 from 7, and bringing down the next digit (2). Each step follows a structured algorithm.

In programming education, Scratch serves as an effective tool to develop Algorithmic Thinking by allowing students to visualize the logical flow of problem-solving. Scratch enables students to construct step-by-step solutions using blocks, making abstract concepts more tangible. The process of arranging blocks in Scratch mirrors the structure of traditional programming, helping students grasp key computational ideas such as sequential execution, looping, and conditional branching more intuitively.

Table 1 illustrates the three fundamental structures of Algorithmic Thinking using Scratch. In the context of studying polygons in mathematics, the procedure for drawing squares is demonstrated through three different programs.

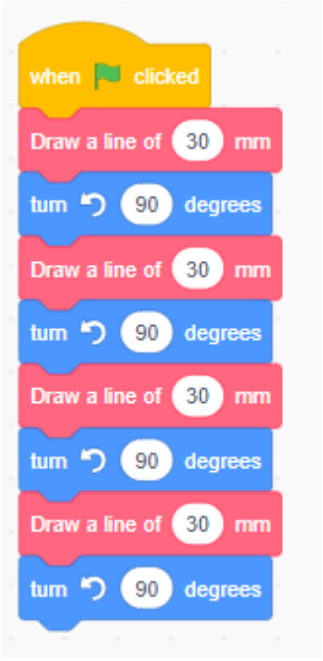
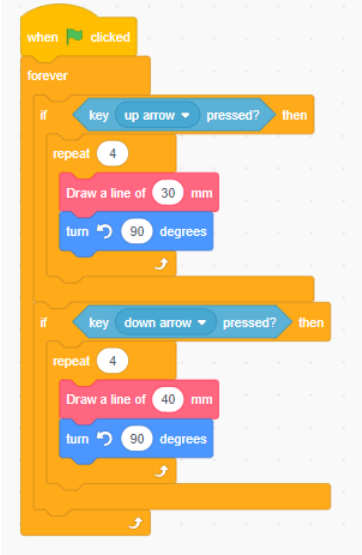
Sequential processing in Scratch involves arranging commands in a defined order to complete a task. For example, drawing a square requires programming four "move forward" and "turn 90 degrees" commands sequentially. The efficiency of the program can be improved using loops, which allow for repeated execution of the same commands, reducing redundancy. Furthermore, conditional branching enables students to create dynamic programs where outputs vary depending on user inputs, such as adjusting the size of a square based on key presses.

Through such exercises, Scratch helps students internalize Algorithmic Thinking concepts in an engaging and interactive manner. By experimenting with different approaches, students develop a deeper understanding of programming logic and problem-solving strategies, ultimately enhancing their computational literacy. By integrating Scratch into educational settings, students are encouraged to think critically and logically, fostering essential skills for their future academic and professional pursuits.

Understanding these fundamental structures helps students develop a systematic approach to problem-solving through programming. By incorporating Scratch into learning activities, students can strengthen their Algorithmic Thinking skills, which can be applied across various disciplines.

Since Scratch is a visual programming language, it does not require learners to master complex languages such as Python and C-Language. However, at some point, students must learn text-based programming languages in order to engage in more advanced programming. In this sense, Scratch represents both an advantage and a limitation: it lowers the entry barrier to programming but does not by itself enable full-scale development. Therefore, after learning programming with Scratch, it is necessary to provide a bridge to text-based programming languages so that learners can advanced to higher levels of programming.

Table 1. Basic structure of algorithmic thinking (Source: Created by the author using Scratch)

Sequences	Loops	Conditions
		

2. 3 Evaluation of Algorithmic Thinking

Ota et al. [11] highlights that "there is little information about the stages of acquiring programming ability." To address this gap, Ota et al. [11] conducted a quantitative analysis of 871 programs created by Japanese elementary school students (grades 4 to 6) and published on the Scratch online community. In an earlier study [11], he developed an evaluation framework for Scratch programs based on the United Kingdom's subject computing learning objectives. The findings revealed that by the fourth grade, students could implement basic programming concepts such as branching, repetition, and variables. By the sixth grade, students were capable of using nested loops, custom blocks, and more complex program structures. A significant improvement in programming ability was observed between the fourth and fifth grades, and further between the fifth and sixth grades. Ota et al. [11] emphasized the need for more detailed analyses of evaluation outcomes and learning behaviors to refine evaluation criteria and improve assessment methods.

Ono and Susono [12] developed two distinct rubrics for assessing Scratch projects. These rubrics were adapted from existing U.S.-based evaluation frameworks, including the "Scratch Rubric for Students" (ScratchED, n.d.) and the "Rubric for Assessing Scratch Projects—DRAFT" (Randall, 2009). To better suit the Japanese educational context, Ono and Susono [12] modified these rubrics and introduced four key assessment categories: Subject Knowledge, Project Design, Programming Technique, and Learning Process. While these rubrics provide structured evaluation criteria, their applicability in real educational settings in Japan remains an open research question. As Ono and Susono [12] suggest, further empirical studies are needed to validate their effectiveness in assessing students' learning progress and computational thinking development. Despite the availability of evaluation frameworks, there remains a gap between theoretical assessments and practical implementation in classrooms.

According to William [13], since CT literature is at an early stage of development, there is no consensus among researchers/scholars in the field and to date, many have been unable to concretely explain what CT is, or how to teach and assess this broad skill set. In addition, CT has been introduced in primary schools worldwide. However, rich classroom-based evidence and research on how to assess and support students' CT through programming are particularly scarce [6]. In order to meet these research gap, evaluating student-generated Scratch programs based on structured rubrics can contribute to refining programming education methodologies and provide valuable insights for educators. Establishing clear and standardized evaluation metrics will be essential for advancing programming education and ensuring its effective integration into school curricula.

2. 4 Practical lessons and research

Miyamoto [14] conducted a lesson using Scratch for approximately 20 sixth-grade elementary school students, comprising a total of five hours of class practice during comprehensive learning time. The lesson was structured as follows: the first hour was dedicated to explaining programming concepts and providing an overview of Scratch, the second and third hours involved hands-on experience with basic Scratch operations, and in the fourth hour, students worked in groups to create original projects using Scratch. During the fifth hour, students continued refining their projects and engaged in a showcase to appreciate their completed works. Throughout these sessions, students programmed various elements, including character movements, sounds, and controls within Scratch.

Fig. 2. Examples of works created by children (Source: Miyamoto, 2018)

As a result of the free work production, all 11 student groups incorporated sequential processing, repetition, and conditional branching in their projects. According to Miyamoto[14], a comparison of pre- and post-test results indicated that students were already somewhat familiar with sequential processing and repetition before the class, which may have facilitated their ability to use these concepts. However, in terms of conditional branching, the study noted that while students were able to use related blocks such as "when the space key is pressed" and "if it reaches the edge, it bounces," they did not fully grasp the underlying concept. Miyamoto [14] concluded that the five hours allotted for this study were insufficient for students to develop a complete understanding of conditional branching. As a future consideration, the study suggested that fostering this skill requires collaboration with other subjects and continuous, long-term learning opportunities.

Following Miyamoto's [14] findings, it is evident that allowing students to create original works through programming can enhance and assess their algorithmic thinking. This study also considers the importance of cross-disciplinary collaboration as highlighted by Miyamoto [14], which will be integrated into future research.

Additionally, Matsui [15] explored programming education through an emphasis on decomposition, a fundamental aspect of computational thinking. In a study involving fifth-grade students in a mathematics course on drawing regular polygons, Matsui [15] incorporated unplugged activities into the 10th lesson of the unit. The students were then guided to use a programming material called "Proguru" in the 11th lesson to create programs for drawing polygons. Unplugged activities allow students to develop computational thinking without the use of electronic devices. For example, in Matsui's [15] study, the teacher used a video demonstrating the movement of a paper cup synchronized with music to help students internalize each discrete action. The activity aimed to reinforce the concept of decomposition and its significance. By the 11th lesson, students successfully applied decomposition to create programs that could draw triangles.

These studies suggest that unplugged activities play an essential role in fostering computational thinking, particularly in the development of algorithmic thinking.

3. Research Questions

Guided by the need to deepen our understanding of algorithmic literacy within STEM education contexts, this study focuses on two primary research questions. The first question addresses the evaluation of algorithmic literacy, emphasizing the roles of subject knowledge, programming design, programming techniques, and the learning process. The second question aimed to investigate the characteristics of students' algorithmic literacy in Japana and Indonesia by examining each evaluation item:

1. How can a newly developed rubric effectively evaluate students' algorithmic literacy in Scratch-based STEM education?
2. What similarities and differences exist between Japanese and Indonesian students' algorithmic literacy?

The development of IT human resources in Indonesia has become an urgent priority. In Japan, programming education has been introduced in schools, where teachers are attempting to implement it through trial and error; however, the body of knowledge in this area remains insufficient. Algorithmic literacy is a critical competency for children worldwide. Consequently, by selecting Japan and Indonesia-two countries with distinct religious and cultural contexts -as the focus of this study, we aim to develop a rubric that can be applied internationally.

4. Methodology

4.1 Research design

Figure 3 illustrates the research design of this study. The study was structured in three phases, beginning with the development of a structured framework based on the previous research by Ono and Susono [12]. Following this, we conducted a total of three intervention lessons in each class, utilizing Scratch, a visual programming learning tool, as the primary platform for instruction. The unit of study focused on drawing polygons in the mathematics curriculum, integrating mathematics, art, and programming as part of a broader STEM education approach.

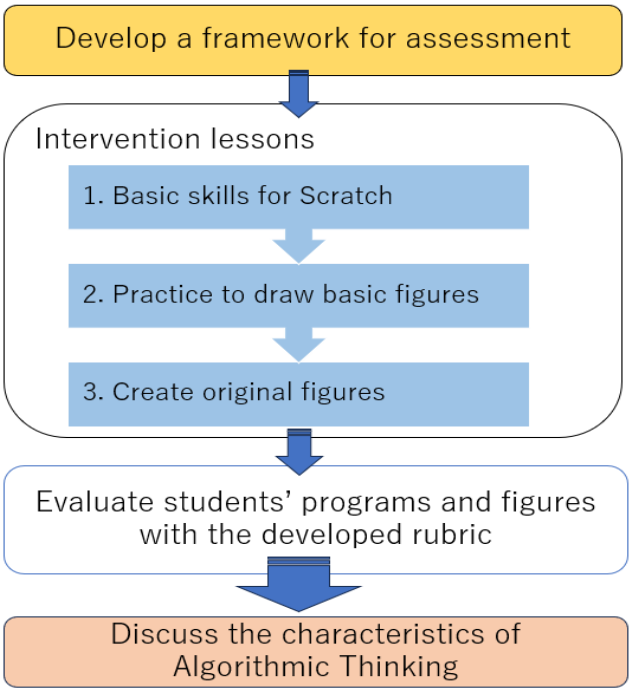


Fig. 3. Research Design (Source: Created by the author)

Table 2 presents the learning objectives and activities for each of the three intervention lessons. In the first lesson, students were introduced to the basic operations of Scratch, along with fundamental concepts of sequential processing and loop processing. Given that most students had never used Scratch before, it was anticipated that terms such as sequential and loop processing would be unfamiliar to them. To address this, the lesson incorporated unplugged activities and visual aids to facilitate students' understanding of algorithmic thinking in an intuitive and engaging manner.

In the second lesson, students engaged in hands-on activities that involved drawing various geometric shapes using sequential and loop processing. This lesson was designed to deepen their familiarity with Scratch operations and reinforce their understanding of how these programming structure functions. Additionally, as students experimented with drawing different figures, they were encouraged to observe the relationships between angles and the number of sides, reinforcing connections to their prior mathematics learning.

The third lesson introduced a more open-ended challenge. Students first tackled level-specific drawing tasks, gradually applying their acquired skills before moving on to designing and creating their own original shapes. This progression—from understanding the concepts in the first lesson, to practicing in the second, and finally applying their knowledge creatively in the third—ensured a systematic and scaffolded learning process. By the end of the intervention, students were expected to not only understand Scratch operations and algorithmic thinking but also to confidently apply them in a meaningful and creative way.

After the intervention, we evaluated the programs and figures created by the students in the final lesson, classifying them into three levels using the rubric developed for this study. This evaluation enabled us to analyze students' understanding and application of algorithmic thinking as demonstrated in their work. Furthermore, we examined the distinctive characteristics of algorithmic thinking among students in Japan and Indonesia by systematically analyzing each component of the structured framework established in this study, which includes subject knowledge, programming design, programming techniques, and the learning process. Through this analysis, we aimed to gain deeper insights into how students from different educational backgrounds approach algorithmic thinking within the context of programming.

Table 2. Intervention lessons (Source: Created by the author)

	Learning Objectives	Students' activities
1	- Learn the basics of Scratch. - Understand how to use sequential processing and loops to draw squares.	- Through unplugged activities, students will explore three key concepts: sequential processing, iteration (loops), and conditional branching. - Create a program in Scratch to draw squares.
2	Apply knowledge to create various polygons.	- Think about how to draw a triangle in Scratch. - Experiment with drawing pentagons, circles, and polygons with more sides, such as hexagons.
3	- Challenge students to draw more complex shapes, such as stars. - Apply knowledge to draw assigned and original shapes.	- Create a program to draw star shapes. - Complete level-based drawing tasks (Levels 1–8). - Use what they have learned to create an original shape through free exploration (Level 9).

4.2 Sample and Data Collection Period

This study targeted a total of 132 students in Japan and 96 students in Indonesia. The participants in Japan included three classes of 6th-grade students (85 students) from public elementary schools in Ibaraki Prefecture and two classes of 5th-grade students (47 students) from public elementary schools in Hokkaido, totaling 132 students. In Indonesia, the participants comprised 30 fifth-grade students from a private elementary school, 32 ninth-grade students from a public junior high school, and 34 eleventh-grade students from a private high school, all located in the Special Region of Yogyakarta.

This study primarily focused on 5th- and 6th-grade students, as the topic of drawing regular polygons, which was central to the intervention, is typically introduced in the 5th-grade mathematics curriculum.

4.3 Development of the rubric

The rubric for evaluating the programs created by students was carefully developed and proposed in this study (Table 3). This rubric aims not only to evaluate students' algorithmic thinking and programming skills but also to capture a broader spectrum of abilities and cognitive processes. To achieve this comprehensive evaluation, the study adopts the concept of "Algorithmic Literacy" rather than the narrower concept of "Algorithmic Thinking." According to Ivan Kalaš [16], algorithmic literacy encompasses "a set of skills that enable an individual to control their behavior and achieve desired outcomes within dynamic environments that include other participants." In this study, these "other participants" represent interactions with teachers and peers, underscoring the interactive and collaborative context inherent in programming education.

Previous research by Ono and Susono [12] provided a foundational rubric to evaluate Scratch programs in school settings, consisting of four categories: Subject Knowledge, Project Design, Programming Technique, and Learning Process. Building on their work, this study adapted the categories to better fit the context of our research, establishing four corresponding categories: Subject Knowledge, Programming Design, Programming Techniques, and Learning Process. This adaptation was undertaken to clearly emphasize practical programming tasks rather than broader project-based activities, thus allowing for a detailed evaluation of the programming logic and sequence structures students used to achieve specific goals, such as drawing geometric figures.

The decision to replace "Programming Knowledge" with "Programming Techniques" highlights the importance of assessing practical programming application rather than purely theoretical understanding. Knowledge can often be evaluated through theoretical assessments such as paper-based tests or questionnaires; however, our study prioritized tangible programming skills demonstrated directly through the students' actual programming activities using Scratch.

Moreover, the rubric employs a clear three-level scale (basic, intermediate, advanced) to differentiate among students' performance in terms of complexity, creativity, and technical proficiency. The developed rubric was specifically designed to assess both the complexity of the shapes created and the effectiveness of the programming structures, including the use of sequential commands, loops, nested loops, and conditional logic. This comprehensive assessment not only evaluates students' understanding of programming concepts but also provides insights into their creative thought processes and problem-solving skills.

Table 3. Rubric for the original figures (Source: Created by the author)

	A:Subject knowledge	B:Programming design	C:Programming technique	D:Learning process
1	Draw figures using 1 angle	The intention of the figures is unclear.	Use blocks of "draw" and "turn"	-Not sufficiently utilized what you learned -Use only 1 figure
2	Draw figures using 2 angles	-The intention of the figures is almost clear. -Create another figure with a figure	Program with sequences and loops	Use 2 figures you learned
3	Draw figures using 3 angles	-The intention of the figure is clear. -Design complicated figures	-Program with sequences and loops -Use "turn-right", "turn-left" and double loops	-Use more than 3 figures you learned -Apply what you learned

Significantly, this rubric is intentionally versatile, allowing adaptation to other programming contexts and educational settings by adjusting its evaluation criteria. Thus, the rubric serves as a practical and versatile tool for assessing algorithmic literacy, extending beyond mere algorithmic thinking. By adopting this broader framework, educators can effectively evaluate the depth and range of skills students acquire through programming education, making the rubric adaptable across various educational disciplines and contexts.

5. Results and Discussion

At the end of the intervention class, the students had 10 minutes to draw original figures with Scratch. Figure 2 shows the original figures and their programs created by the students in Japan, and Figure 3 shows the original figures and programs created by the students in Indonesia. Since some of the data storage did not work well, the number of samples that could be collected this time was 73 samples in Japan (Grade 5:32, Grade 6:41) and 53 samples in Indonesia (Grade 5:11, Grade 9:26, Grade 11:16). Some students were not able to save the programs they had created, so some data could not be collected even though the students had created the programs.

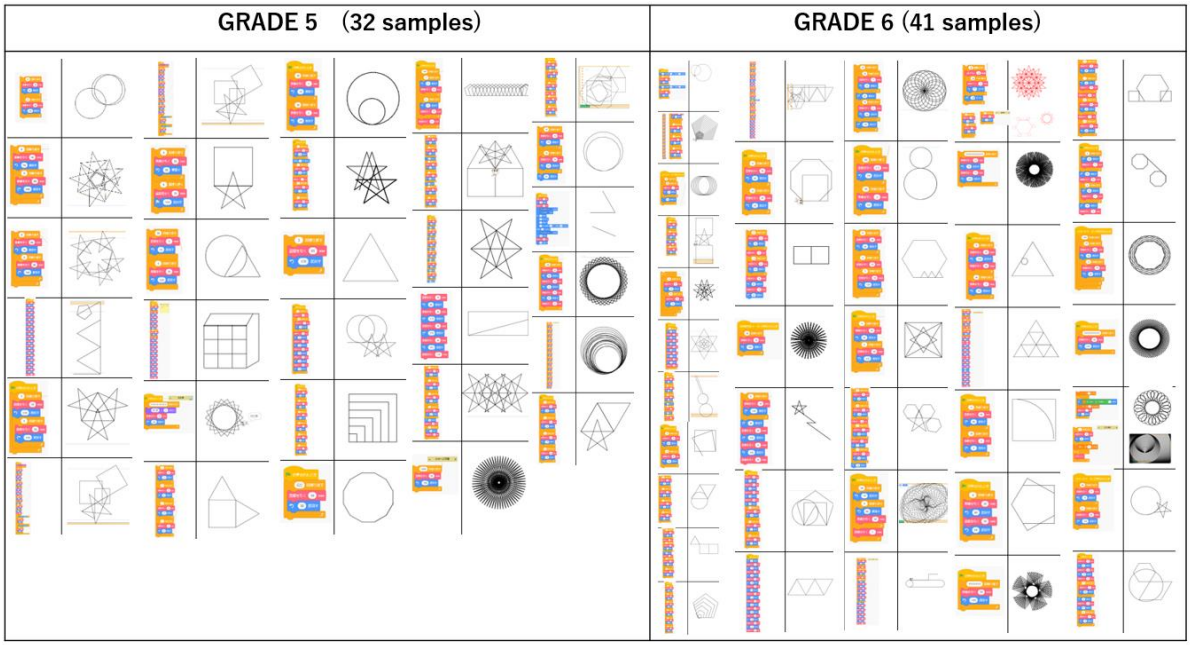


Fig. 4. Scratch works by Japanese students

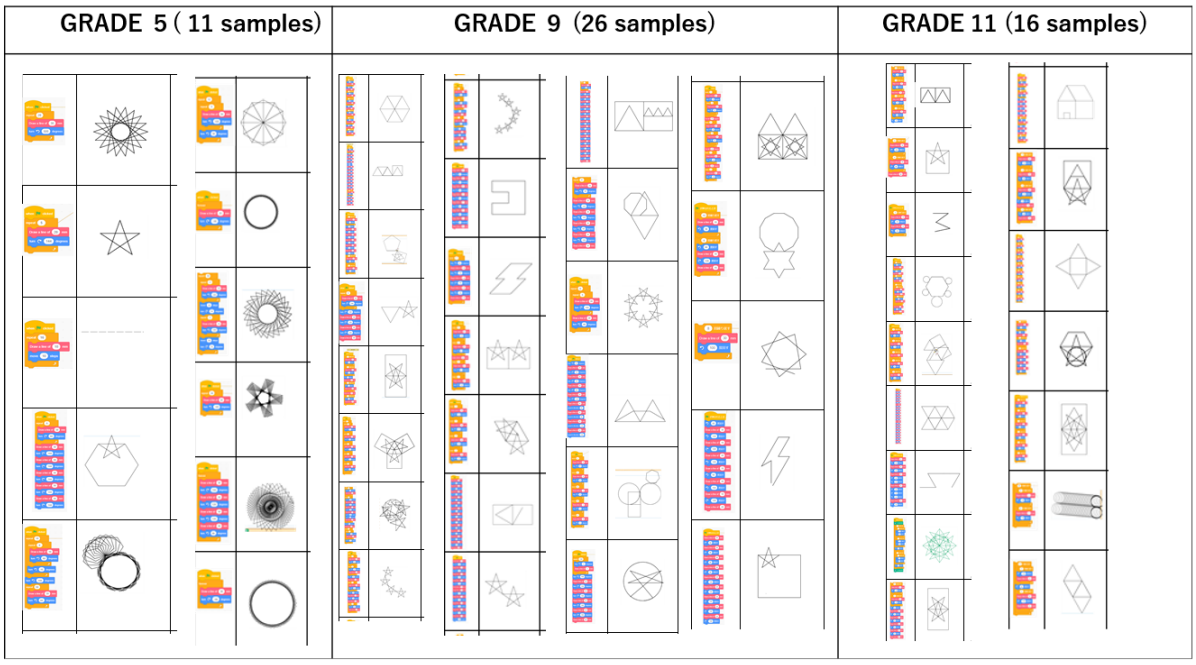


Fig. 5. Scratch works by Indonesian students

The rubric assessment provided insights into students' algorithmic literacy by examining four key areas: Subject Knowledge, Programming Design, Programming Techniques, and Learning Process, as conceptualized by Kalaš's definition of algorithmic literacy [16]. Table 4 and table 5 show the result of the rubric assessment for each.

Table 4. Result of the rubric assessment (Japan, n=73)

	A:Subject knowledge	B:Programming design	C:Programming technique	D:Learning process
1	21%	29%	9%	30%
2	51%	29%	43%	44%
3	27%	43%	49%	26%

Table 5. Result of the rubric assessment (Indonesia, n=53)

	A:Subject knowledge	B:Programming design	C:Programming technique	D:Learning process
1	23%	25%	19%	36%
2	43%	42%	38%	36%
3	34%	34%	43%	28%

First, examining the results in detail, we found notable patterns across the rubric categories. In Japan (N=73), the proportions of students receiving the highest rating (Level 3) in each category were as follows: Programming Technique (49%), Programming Design (43%), Subject Knowledge (27%), and Learning Process (26%). Similarly, in Indonesia (N=53), the proportion of students at Level 3 was highest in Programming Technique (43%), followed by Programming Design and Subject Knowledge (both 34%), and Learning Process (28%).

Among the four rubric categories, "Programming Technique" consistently showed the highest proportion of students achieving Level 3. This indicates that many students effectively utilized both sequential and loop programming structures, skillfully creating original shapes by integrating multiple blocks, including sophisticated maneuvers such as changes in rotation direction of sprites. Such findings suggest that students not only understood the fundamental concepts of sequential commands and loops but also developed advanced programming competencies, allowing them to produce complex and unique designs. This demonstrates their growing proficiency in controlling programming behavior and achieving desired outcomes, essential aspects of algorithmic literacy.

In the "Programming Design" category, students' works were evaluated based on clarity and complexity of their intended designs. A significant number of students, particularly at Level 3, successfully communicated clear design intentions and demonstrated considerable creativity by developing intricate and original geometric figures. These outcomes suggest that many students in both Japan and Indonesia established clear goals and effectively translated these goals into executable Scratch programs. The ability to plan and structure complex figures highlights a deeper cognitive engagement, suggesting that the students actively employed goal-setting, decomposition, and iterative refinement through trial and error, key components of algorithmic literacy.

In the "Subject Knowledge" category, the evaluation focused on the application of mathematical concepts, specifically angles of geometric figures. In both Japan and Indonesia, Level 2 was the most frequently observed level. This suggests that students generally found it manageable to integrate two distinct angles within their original figure designs but experienced greater challenges incorporating three or more angles. This pattern could reflect variations in their level of understanding or confidence in applying geometric concepts, potentially influenced by prior mathematical instruction and individual interest levels. Thus, mathematical subject knowledge seems to form an accessible foundation but indicates room for further development and application to more complex geometrical reasoning within programming tasks.

The "Learning Process" category evaluated how effectively students utilized multiple learned shapes in their programming. Level 2, involving the utilization of two figures, was most common. This result might reflect a balance between cognitive load and the complexity of task requirements within the limited timeframe provided. Students were generally comfortable integrating two previously learned figures but faced challenges when required to combine more complex or multiple figures. Enhancing students' capacity to apply learned content in more complex and integrative ways requires additional supportive instructional strategies and extended practice.

Overall, the findings align with the theoretical framework based on algorithmic literacy proposed by Kalaš [16]. Students demonstrated goal-oriented behavior by clearly identifying tasks and progressively refining their programs through iterative processes of decomposition, abstraction, pattern recognition, trial and error, and reconstruction. These iterative processes reflect the core aspects of algorithmic literacy, where students dynamically adjust their approaches in response to feedback and interactions with peers and teachers.

The analysis underscores the critical interplay between algorithmic thinking and subject knowledge, highlighting that neither skill alone sufficiently supports comprehensive problem-solving. Rather, the integration of subject knowledge with algorithmic thinking skills through practical programming activities is crucial. Hence, this study emphasizes the importance of simultaneously fostering strong foundational knowledge and enhancing students' practical programming capabilities, promoting holistic algorithmic literacy essential for effective problem-solving in varied educational contexts.

6. Conclusion

This study aimed to analyze and evaluate students' Scratch works (programs and figures) using a carefully developed rubric based on the concept of "Algorithmic Literacy." By classifying students' work into three performance levels, we captured differences in the number of angles used, the intention and complexity of the figures, the programming blocks utilized, and the frequency with which learned shapes were applied. Consequently, a comprehensive and versatile rubric was developed.

Addressing the first research question—"How can a newly developed rubric effectively evaluate students' algorithmic literacy in Scratch-based STEM education?"—the rubric created in this study demonstrates an effective method for evaluating algorithmic literacy, covering knowledge, creativity, programming techniques, and learning attitudes. The versatility of the rubric allows it to be adapted across various subjects, promoting a holistic evaluation of programming education by capturing different aspects of students' cognitive, creative, technical, and attitudinal dimensions.

Concerning the second research question—" What similarities and differences exist between Japanese and Indonesian students' algorithmic literacy?"—the following insights were obtained. First, students exhibited a high proficiency in programming techniques, particularly in applying sequential and loop structures. Second, their approach to programming tasks reflected the problem-solving method outlined in the theoretical framework, including setting clear objectives, decomposition, trial and error, abstraction, pattern recognition, and reconstruction. Third, it was confirmed that a solid foundation in subject knowledge and creative thinking significantly supports effective algorithmic thinking, underscoring the importance of integrating subject matter expertise and creative skills in programming education. In this study, no significant differences were observed in the algorithmic literacy of students in Japan and Indonesia.

However, this study has several limitations. The number of sample data collected was restricted due to technical difficulties in saving students' Scratch projects, particularly in Indonesia. It is essential to ensure proper management of students' Scratch accounts and passwords for future research to avoid data loss. Additionally, further research should involve secondary and high school students in Japan to broaden the applicability and robustness of the findings.

Furthermore, to enhance the validity and reliability of the developed rubric, future research should incorporate multiple evaluation methods, such as triangulating rubric assessments with qualitative feedback from teachers and peer evaluations. Further studies should also explore diverse subject contexts to test and refine the adaptability of the rubric.

In conclusion, the rubric and conceptual framework proposed in this study provide a foundation for comprehensively assessing algorithmic literacy, encompassing knowledge, creativity, technical proficiency, and learning processes. Future research should expand on the diversity of subjects and educational levels to refine the rubric further and validate its applicability across broader educational contexts. Additionally, longitudinal studies examining the long-term impact of programming education on students' algorithmic literacy would be beneficial, enhancing our understanding of how sustained programming education influences cognitive and creative development.

References

[1] Ministry of Education, Culture, Sports, Science and Technology (MEXT). General provisions. Course of Study for Elementary Schools. 2017.

[2] Google, TEMASEK, BAIB&COMPANY. e-conomy SEA 2019. Google TEMASEK BAIB&COMPANY. 2019.

[3] Hilyana F. Shoufika, et al. STEM-based Digital Assessment Application for Elementary School Teacher Education Students, Department of Primary Education Teacher, Universitas Muria Kudus, Gondang Manis Bae Kudus. BIO Web of Conferences 117, 01026 (2024) https://doi.org/10.1051/bioconf/202411701026. ICoLiST 2023. Central Java, Indonesia. 2024.

[4] Hardika Dwi Hermawan, et al. Implementation of ICT in Education in Indonesia during 2004-2017. Information and Technology Studies Faculty of Education The University of Hong Kong HKSAR, P.R.China . 2018 International Symposium on Educational Technology. DOI 10.1109/ISET.2018.00032. 2018

[5] William H. STEWART and Kwanoo BAEK. Analyzing Computational Thinking Studies in Scratch Programming: A Review of Elementary Education Literature, Hankuk University of Foreign Studies, Korea. University of Southern California, USA. International Journal of Computer Science Education in Schools, March 2023, Vol. 6, No. 1 ISSN 2513-8359. DOI: 10.21585/ijcses.v6i1.156. 2023

[6] Janne FAGERLUND et al. Assessing 4th Grade Students' Computational Thinking through Scratch Programming Projects. Department of Teacher Education, University of Jyväskylä, Jyväskylä, Finland. Informatics in Education, 2020, Vol. 19, No. 4, 611–640 © 2020 Vilnius University, ETH Zürich. DOI: 10.15388/infedu.2020.27

[7] Ministry of Education, Culture, Sports, Science and Technology (MEXT). Ability to develop through programming education. Guidance of programming education in elementary school. [3], 11－12. 2020.

[8] Wing, J. Computational Thinking, http://www.cs.cmu.edu/afs/cs/usr/wing/www/Computational_Thinking.pdf, ACCESS.2023.08.23.2007.

[9] Mandoye. N. Teaching basic problem decomposition and algorithm design skills, Tuskegee University. https://serc.carleton.edu/teaching_computation/workshop_2019/essays/231291.html, ACCESS.2023.06.03. 2019.

[10] Adem Doğani, Algorithmic Thinking in Primary Education, Kahramanmaraş Sütçü İmam University. International Journal of Progressive Education, 2020, Vol.16. https://doi.org/10.29329/ijpe.2020.268.18, 2020.

[11] Ota et al. Quantitative Analysis for Acquisition of Children's Programming Skills: Scratch Programming of Grade 4–6. Information Processing Society of Japan. Education and Computer. Vol.5 No.3 35–43. 2019.
[12] Ono Eri & Susono Hirotoshi. How to evaluate students' Scratch projects in Japan, JSSE Research Report. Vol. 31. No. 8. 2017.
[13] William, H. Analyzing Computational Thinking studies in scratch programming : A review of elementary education literature. International Journal of Computer Science Education in Schools, March 2023, Vol. 6, No. 1. 2023.
[14] Miyamoto, K. & Kawano, S. Practice and verification of the computer programming class with scratch in an elementary school, Journal of the Japan Society of Technology Education. Vol. 60. No. 1. 19－28. 2018.
[15] Matsui Touru, YokoyamaTakamitsu, Saitou Youko, Imai Sena. A trial of programming education. Faculty of Cultural Creation, Gifu Women's University. 2019.
[16] Kalaš, I., 2011. Spoznávame potenciál digitálních technológií v predprimárnom vzdelávaní. Ústav informácií a prognóz školství, Bratislava.
[17] Bando Tesuya, et al, Proposal for Evaluating the Programming Thinking Process in Lower Elementary School Grades. The Japan Society of technology Education. Vol.63(1), pp.111－119, 2021.
[18] Cabinet Office. Science and Technology Policy. Council for Science, Technology and Innovation, Cabinet Office. 2019. https://www8.cao.go.jp/cstp/english/society5_0/index.html, ACCESS.2023.06.22.
[19] Dainihontosho. Interesting mathematics web. Dainihontosho. https://www.dainippon-tosho.co.jp/web/sansu/065scratch/index.html, 2019, ACCESS.2023.05.24.
[20] Linda. L. Hello Ruby. (Tori. Y, Trans.). Shoeisha. 2016.
[21] Ministry of Education, Culture, Sports, Science and Technology (MEXT). The model of the programming education in primary school, 2016.
[22] Okahana Kazuki, et al. Computational approach in elementary school mathematics aiming meta-cognition of analog geometry, Japan Society of Digital Textbook. ISSN 2432-6127 Vol.7, 2018.
[23] Saito Kazuhisa. Regular polygon, The point of new mathematics teaching, Toyokan Publisher. 1993.
[24] Soma Kazuhiko. et al. Interesting mathematics Grade 6, Dainihontosho. 2020.
[25] Tatiana Havl ásková et al. Applications that help develop algorithmic thinking, University of Ostrava, Fráni Šrámka 3, Ostrava, Czech Republic, 17th International Conference e-Society. 2019.

Authors' Profiles

Hiroyuki Endo, Master, completed his Bachelor's degree in English language and literature at Senshu University, Kanagawa, Japan. He started to work as an English teacher at a secondary school in Ibaraki in Japan. He joined the Japanese volunteer program of Japan International Cooperation Agency (JICA). As a volunteer, he facilitated and promoted sports and cultural activities for children and adolescents in the Community Development Center in Djibouti. He began to work as a teacher in a primary school in Ibaraki in Japan. In total, he has over 10-year experience as a teacher. He received Master's degree in Education from Naruto University of Education in Japan. His current job is consultant for Grant Assistance for Grassroots Human Security Projects (GGP) in the Japanese Embassy in Kinshasa, the Democratic Republic of the Congo. His research interest includes Computational Thinking, Mathematics Education, Primary School Education.

Satoshi Kusaka received the M.A. degree in education and development from the University of Leeds, Leeds, U.K., and the Ph.D. degree in education from Hiroshima University, Hiroshima, Japan. His major field of study was mathematics education. He has worked on various international education projects, including collaborations with Japan International Cooperation Agency (JICA), and served as a research collaborator for the OECD's Global Teaching Insights. He co-authored Elementary School Mathematics (Tokyo, Japan: Gakko Tosho) and held editorial roles for educational publications. He is currently an Associate Professor at Naruto University of Education, Naruto, Japan. His current and previous research interests include metacognitive skills in mathematics learning and the endogenous development of mathematics curricula. Dr. Kusaka has received multiple research grants, including those from Japan's Grants-in-Aid for Scientific Research, to explore the revision of mathematics curricula in various countries. He also serves as the Chief Editor for the English version of the Gakko Tosho elementary mathematics textbook series.

Rully Charitas Indra Prahmana began his academic career in 2009 at Universitas Muhammadiyah Makassar and was appointed as a Full Professor of Mathematics Education at Universitas Ahmad Dahlan in 2022. His academic endeavors are primarily centered on the integration of cultural contexts into the Realistic Mathematics Education framework, culminating in the formulation of the Ethno-Realistic Mathematics Education approach. This pedagogical innovation combines the principles of ethnomathematics and RME to promote mathematics learning that is both culturally relevant and contextually meaningful. His research interests lie predominantly in ethnomathematics—examining the relationship between cultural practices and mathematical concepts—and in realistic mathematics education, which advocates for the application of mathematics to real-world contexts.

Nur Robiah Nofikusumawati Peni began her academic career in 2021 and was promoted as Assistant Professor at Universitas Ahmad Dahlan in 2023. Throughout her career, she has focused on integrating cultural context into mathematics education, a field known as ethnomathematics, and has also contributed significantly to the development and analysis of mathematics curricula through a realistic mathematics education approach. She completed her Bachelor's degree in Mathematics Education at Universitas Ahmad Dahlan, Indonesia, in 2012. She then pursued advanced studies in Japan, where she earned her Master's degree, in 2017, and Doctorate, in 2021, in Mathematics Education from Hiroshima University. Her research interests lie primarily in ethnomathematics, where she explores the intersection of culture and mathematics, and in realistic mathematics education, which emphasizes the application of mathematics to real-world contexts. She is also deeply involved in the development and analysis of mathematics curricula, aiming to enhance the quality of mathematics education at various educational levels.

How to cite this paper: Hiroyuki Endo, Satoshi Kusaka, Rully Charitas Indra Prahmana, Nur Robiah Nofikusumawati Peni, "Assessing Algorithmic Literacy through Scratch Programming: A Comparative Study in Japanese and Indonesian STEM Education", International Journal of Education and Management Engineering (IJEME), Vol.15, No.5, pp. 1-12, 2025. DOI:10.5815/ijeme.2025.05.01