International Journal of Learning, Teaching and Educational Research
Vol. 19, No. 8, pp. 98-116, August 2020
https://doi.org/10.26803/ijlter.19.8.6

Contextualising Computational Thinking: A Case Study in Remote Rural Sarawak Borneo

Nur Hasheena Anuar, Fitri Suraya Mohamad and Jacey-Lynn Minoi
Universiti Malaysia Sarawak, Kota Samarahan, Sarawak, Malaysia
https://orcid.org/0000-0002-3677-8514
https://orcid.org/0000-0003-4460-8061
https://orcid.org/0000-0002-9464-2631

Abstract. The paper describes an exploratory case study on novice indigenous children's learning characteristics as they learn Computational thinking (CT) competencies, such as abstraction, decomposition, and algorithmic thinking. It employs a quasi-experimental research design with pre-test and post-test instruments. Twenty-two children of an underprivileged Penan community living in a remote village in Sarawak Borneo participated. Through the study, they learned Computational thinking skills using localised instructional strategies, with Scratch™ as their tool to programme. The study used observational field notes, comprehension checks, and participants' learning products as primary data sources. Findings showed that indigenous children's learning characteristics were primarily 'learning-by-making', collaborative, highly motivated, playful, curious, and imaginative while they attempted to learn Computational thinking. The intervention (treatment) group performed marginally better than the control group in the pre-test and were substantially better in the post-test performance. Findings illustrate a direction in which novice indigenous children could learn and be informed about Computational thinking practices and skills through a mix of game-based learning, collaborative learning, problem-based learning, and project-based learning. Findings also revealed how participants appeared to have gained self-confidence, illustrated creativity on task and were self-critical throughout their participation in the study.

Keywords: Computational thinking; Remote classroom; Indigenous children; Penan community; Case study

1. Introduction

Developing computational thinking skills from a young age has become crucial in today's setting as it is seen to fulfil the demand of social and global economic growth (Wing, 2010; Grover, 2018; Haseski, Ilic & Tugtekin, 2018). At the macro level, ensuring an increase of uptake among young students to pursue STEM education, and later careers, has become a critical agenda in Malaysia's Education system. ‘STEM For All’ initiative is a national strategy by the Malaysian Ministry of Education (MOE) to enhance the participation in STEM and STEAM (Science, Technology, Reading, Arts and Mathematics)” (Mustafa, 2019). The initiative was prompted by the decrease of STEM students to a mere 44% in 2019. Another STEM movement in Malaysia is the *mydigitalmaker* initiative, spearheaded by Malaysian Digital Economy Corporation (MDEC). MDEC leads Malaysia’s digital economy. The programmes and activities are designed in collaboration with private and public academia. They have collectively impacted 1.2 million students (Malaysia Digital Economy Corporation, 2019). Under *mydigitalmaker* initiative, “Hour of Code” programme has been organised countrywide to cultivate programming interest and skills among young students.

In line with current demands for 21st Century Learning, MOE has revised its STEM initiative and Digital Economy, and both standard curricula of primary and secondary schools, as reported in the Malaysian Education Blueprint 2013-2025 (Ministry of Education, 2013). Beginning 2017, Primary One students have begun learning basic computer skills such as placing text and images into Word processors, making slides for the presentation and searching information via an Internet browser (Kementerian Pendidikan Malaysia [KPM], 2014; Ling, Saibin, Abidin & Aziz, 2017). Despite the aspirations of the newly revised curriculum, Malaysian students have not advanced to application-level due to lack of technology access and exposure. They are still regarded as digital technology consumers.

The term ‘computational thinking’ became popular when Papert first introduced LOGO programming to young children for training their procedural thought (Grover & Pea, 2013). Since then, educators have often begun teaching and implementing CT in K-12 by introducing computational skills and computational engagement (Kafai & Burke, 2017; Weintrop et al., 2016). Moschella (2019) defined computational thinking as mental abilities to perform problem-solving, abstraction, using algorithms, restructuring processes, reformulating problems, and implementing solutions. Computational thinking skills are now considered as a fundamental skill, much like reading, writing, and counting while addressing real-world issues (Wing, 2006).

A fundamental concern is that educators are unmotivated to adopt computational thinking strategies in the classroom. Many educators generally are not inspired due to time limitations to using machine technology, coupled with the lack of pedagogical skills (Selby, 2014). Despite these challenges, the practices of computational thinking should be carried out with specific methods and tools as early as possible (Moschella, 2019). Similar context to this study, *My Elephant Friend*, was used as a game tool for education in programming and computational thinking to educate young novices in India (Unnikrishnan et al., 2016).

Some of the popular tools used by teachers in schools to teach computational thinking and programming skills, include CS Unplugged, Code.org curricula, and Scratch (Israel, Pearson, Tapia, Wherfel & Reese, 2015). They mainly involve interactive and computerised exercises that facilitate the comprehension of children's computing concepts. However, there is a dearth in the literature about how teachers at remote rural locations dealt with the teaching of computational thinking skills. To date, there is no research on how the teaching of computational thinking is localised to a context, especially for remote rural primary schools.

Hsu, Chang and Hung (2018) suggested that project-based learning, problem-based learning, collaborative learning, and game-based learning as the most effective teaching techniques used in early school years to facilitate computational thinking. They further explained how project-based learning approach enables learners to learn CT skills in positive attitudes. As a project-based approach is a group-based activity, it may motivate the learners to work with their peers in achieving the goals. Through the problem-based approach, students start by recalling prior knowledge related to a situational problem and apply CT skills into it. The main advantage of these learning strategies is that it enhances learners' understanding of new knowledge while building a positive attitude towards learning and becoming self-directed learners (Cantillon, Hutchinson, & Wood, 2003; Tseng, Chang, Lou & Chen, 2013). Inexperienced learners may face difficulty understanding concepts in the absence of problem-solving and metacognitive techniques (Blumenfeld et al., 1991).

It was also suggested that learning CT skills through game-based approach increased students' motivation and the 'flow' of a learning experience (Kazimoglu, Kiernan, Bakon & MacKinnon, 2012). In the present study, an unplugged game was developed by adopting a game-based approach as it was intended to help students to understand algorithms concept in a playful manner. With Scratch™, novice learners are expected to encounter less difficulty as it supports low-level programming and high-ceiling learning environment. Learners could create and be creative without being evaluated by the programme or receive syntax error messages from Scratch™.

The focus of this paper is to present a study on learning characteristics of a group of young novice underprivileged indigenous students, as they learn computational thinking skills for the first time through a deliberately localised instructional strategy. We targeted three CT skills: abstraction, decomposition, and algorithmic thinking, as they have the closest affinity to what primary school students could use in game-based activities and programming tasks, based on their environmental setting and knowledge readiness. The CT capabilities in this study refer to cognitive skills rather than practical skills. The study incorporates an adaptive model for young novice children to learn CT concepts and practices through a redesigned CT activity. The data collected are meant to improve the way children are taught Computational Thinking in a remote learning environment.

Taking a cue from Hsu et al.'s findings, the study also adapted the problem-based approach, in which drawing and colouring tasks were localised to a situational problem. Kafai and Burke (2014) stated that learning how to programme and skills to interact with others are crucial steps in broadening computational participation. However, computational participation may be challenging for students who come from low socio-economic backgrounds and reside in distant schools due to their restricted Internet accessibility and technology usage (Mohamad, Yeo, Abd Aziz & Rethinasamy, 2010). Thus, the process of acquiring CT skills for students with various abilities, interest, and different learning paces, especially novices and underprivileged, maybe highly challenging. Right from the beginning, we were aware of the challenges to conduct the study with the target group. Participants worked in a team and helped each other when working on assigned Scratch™ projects. The learning tasks designed for the study seemed to have enabled them to build their confidence in dealing with digital technology.

2. Computational Thinking in the Classroom

Computational thinking has been associated with learning of computer science. For decades, researchers and educators had teamed up to build a computational thinking model and define its definition, engage, and assess children's capability to think computationally. Wing (2006, 2008, 2010) described CT as fundamental mental activity and skills to both human and machine in solving problems including logical thinking, analytical thinking, mathematical thinking, engineering thinking, scientific thinking, and algorithmic thinking to find a solution. Until today, there is no conclusive definition of CT for all as educators have various learning demands and objectives to achieve.

In middle- and high- schools of Western countries, CT is taught through a flexible approach in higher education (National Research Council, 2011). For instance, the CS Principles course (http://www.csprinciples.org/) attempts to incorporate the concept of computing and practices of computational thinking such as the development of digital artefact for creative expression targeted on high school students (Grover et al., 2013). Through the course, students were reported to be able to master computational skills and solve problems, communicate new ideas, and collaborate with others. However, at the point of writing, there is no known model which has been developed for teachers in remote rural schools to teach primary school students about Computational thinking skills.

According to Papert and Harel (1991), Constructionism is a 'learning-by-making' approach. It is a suitable approach to learning computational thinking skills. Robles et al. (2017) and Hsu et al. (2018) found that teaching CT skills through constructionism approach can implement with or without computers. For instance, CS Unplugged (https://csunplugged.org/) is a module that is used to teach young learners about computer science topics without using computers. The topics consist of CT elements implement through intriguing games and paper-based materials. The advantage of Constructionism and unplugged approaches are cost-effective and suitable for all age (Durkin et al., 2018). For example, unplugged cybersecurity modules and activities developed by Durkin and his colleagues (2018) to promote engagement and understanding in K-12 educators of computer science issues. The result is promising as K-12 teachers are highly engaged with their students and allow rapid implementation into their classroom.

Asad, Tibi and Raiyn (2016) admit that learning programming is not easy for children, as it involves the use of computational and algorithmic thinking. Ben-

Ari (as cited in Asad et al., 2016) suggested that Scratch supports the development of CT skills through low-level programming with a highly supportive learning environment. Scratch is a visual programming tool that helps learners to retain information better when dealing with abstract concepts and ideas. It is designed to support a wide range of learners, including novice, low digital literate, and adult users (Lin, 2015), also, highly engaging. Learners will not encounter syntax error when using Scratch for building codes blocks. A language selection function is included in Scratch, which enables students to choose their language preferences. Hence, researchers suggest that Scratch is the most suitable tool for novice learners to learn computational thinking skills.

Moschella (2019) discussed the teaching of three programming elements (i.e., abstraction, decomposition and use of algorithms), and their correlation with young learners and methods adopted for computational thinking for primary school children. She used a systematic literature review to extract standard tools commonly used by educators to teach algorithmic thinking, abstraction, and decomposition at the primary school level. She also found that and Scratch™ and game-based learning were widely used to teach algorithmic thinking.

Game-based learning (GBL) is a part of the wider gamification approach that use game mechanics or techniques to learn concepts in a serious but playful manner to engage learners (Paniagua & Istance, 2018). Kazimoglu et al. (2012) found that learning CT skills through GBL approach increased students' motivation and the 'flow' of a learning experience. It helps students to understand concepts quickly and discover how algorithms work. Scratch™ is widely used by young and adult learners to create creative computational artefacts such as Pinball and Shooter - games that have a similar approach to GBL.

Problem-based learning (PBL) is a vital process for students to acquire knowledge and skills (Baharom & Palaniandy, 2013). They had claimed that students started to learn from problems that occur in their everyday life, which creates prior knowledge. Slowly, it builds the foundation to obtain new ideas and confidence in tackling existing problems. Hsu et al. (2018) found that project-based learning is the most relevant strategy to introduce computational thinking skills at schools. Project-based learning is a constructivist approach whereby students learn how to engage and solve problems with a scientific design within a group learning environment (Tseng et al., 2013). Students who participated in project-based learning will tend towards positive in learning. Using a project-based learning approach, students can advance their development of CT skills and develop a better understanding of the skills. Hsu et al. (2018) supported problem-based learning is a widely used method in primary school to teach and develop CT skills of different subjects. It may involve students to develop the program design as their project-based assignment.

Collaborative learning has also been found to be useful in CT activities as students would discuss similar problems, communicate with common strategies and plan for solutions. According to Perković, Settle, Huang and Jones (2010), there are four dimensions of collaborative learning which are circumstances, interactions between group members, learning systems, and outcomes of collaborative learning. It allows students to work together, to exchange ideas, and interacting throughout the learning process. It also helps them to develop teamwork and interpersonal skills when participating in computational activities.

While most studies reviewed were conducted in developed nations, it is evident that the challenges in teaching Computational thinking skills in classrooms as diverse as Malaysia would be different. In the context of this study, teachers in a remote school may find teaching computational thinking skills to indigenous children exigent. While the push from the national curriculum toward a more progressively global trend in digitally integrated ways of learning, teachers in remote schools seem to affix to their rural learning environments. To date, there has not been any known research done to understand how indigenous children at Sarawak learn computational thinking skills such as algorithms, abstraction, and decomposition, especially in their social context of learning. Localising a relatively foreign knowledge such as computational thinking would impose a unique undertaking, as the rural students most likely do not grow up with digital tools nor simulated environments which would encompass the need to learn abstraction, decomposition and use of algorithms.

Yadav, Hong and Stephenson (2016) claimed that offering computer science subject in a rural school would serve a massive challenge as school administrators and the school board would not see it as a priority subject. In other words, computer science topics would not be considered in a school examination subjects' list. The rationale for excluding computer science subjects in remote schools may be due to the lack of resources to teach the subject (such as expertise from IT background, digital equipment, and Internet connectivity). It was suggested that such subjects could still be offered through the use of unplugged (non-computer) materials and practical computer tools such as Scratch™.

The present study was designed to introduce localised instructional strategies to teach computational thinking skills such as algorithmic thinking, abstraction, and decomposition for a group of young indigenous Penan children in remote rural Sarawak. Findings from the study would not be generalised to other indigenous communities in the country as they may embrace dissimilar social, emotional, mental, and spiritual values.

3. Methodology

The study employs a case study approach to observe, measure and analyse behavioural and learning impact among 22 young indigenous students who live in a remote rural location in Sarawak Borneo. Field observations were carried out to document interactions, behaviours and competencies of the school students and their teachers. A set of localised instructional strategies were then carefully selected and developed to introduce Computational thinking concepts to the participants. They were divided into two groups; one group received an instructional intervention, where an unplugged coding game was designed to facilitate the comprehension of Computational thinking concepts. Both groups were introduced Scratch as a tool to produce learning tasks which encapsulate an understanding of how computational thinking works.

The location of the study is a remote rural village of just over 600 indigenous Penan, a once nomadic hunter-gatherer tribe which has settled in the area in the past twenty years (Zaman & Falak, 2019). In 2010, it was estimated about 16,281 of Penan population had settled in Sarawak (Zaman et al., 2016). For Penan students, they prefer to speak in Penan language as their first language in the school, instead of the national language, Bahasa Melayu. It is common for rural communities such as these to use their language for communication due to the strong cultural influence and attachment in the community. From their formal education, they gradually learn to use Bahasa Melayu as the primary communicative language with anyone outside of their tribe (Law, 2018).

The school was built on a wooden structure, and it houses six classrooms and two administrative rooms. All students come from the same village, and they all live at the school's hostel. They eat, sleep and study together, and undeniably their camaraderie with each other was formidable. During the time of the study, there are 56 students attended its Primary School. After they complete Primary Six, they would leave the community to live in a boarding school about one hour away by boat. There is no gravel road access to the village; the community rely on a river as their primary transportation route and for providing basic water needs. The electrical supply was provided by diesel-fuelled single-user generators and a small hydro-electric unit which was donated by the Government of Japan.

Before the study commenced, we sought permission from the Research Ethics Committee of University Malaysia Sarawak, the Sarawak State Education Department, and the Educational Research and Planning Department of the Ministry of Education in Putrajaya. We have addressed all four main elements of human and ethics research – respect for persons, respect for personal autonomy, beneficence and justice, throughout the conduct of the research. On top of the seeking ethics approval and permission from official channels, we also sought permission from the community elders using an established cultural protocol which was developed by Phoa (2009) (Minoi et al., 2018).

The study was conducted over three separate visits to the school and the community. In the three visits which took place over 12 months, we observed and interacted with the community members and leaders, and with the schoolteachers and principal.

Twenty-two children from the school volunteered to participate in our study. They attended Primary Four (15) and Primary Five (7) classes, aged 10 and 11, eleven boys and eleven girls. None had any experience in programming and no background knowledge in computational thinking skills. We describe them as active, obedient, shy, and possess a high level of curiosity. Through a subject taught at school, called Information and Communication technology, they have learned about the critical functions of a computer, and they were familiar with word processing and digital multimedia. Unfortunately, they have minimal access to computers because of the lack of technology tools at the school and their own homes. The students mostly preferred to play outdoor sports such as football and badminton, instead of sedentary games.

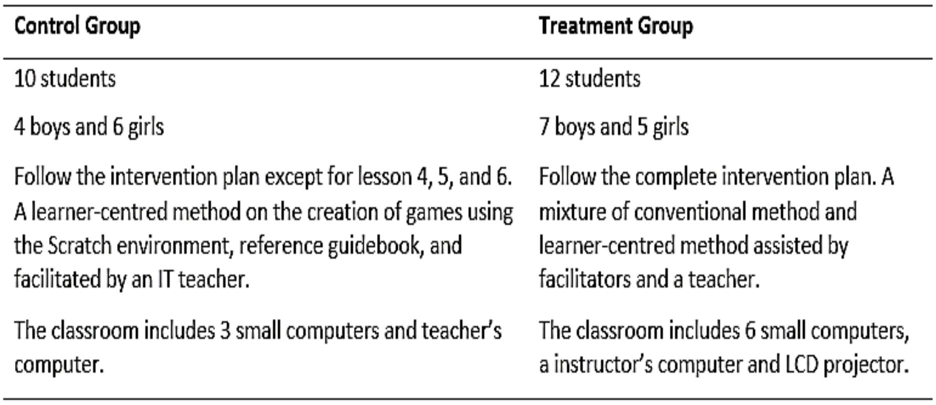
Our initial interactions and analysis enabled the design of the localised instructional strategies, which we eventually used in the introductory sessions on Computational thinking (Minoi et al., 2018).

Two key activities were designed and developed for the study. One was called "unplugged coding game" and the second one was focused on introducing Scratch™ as a tool for programming.

Participants are divided into two groups - the control and treatment groups. The control group has ten students comprised of four boys and six girls, while the treatment group had 12 students, seven boys and five girls. For the setting (see Table 1), both groups shared one classroom space during Lesson 1, 2, 3, and 7 as they needed to collaborate in the activities. During Lesson 4, 5 and 6, both groups are separated into two classroom spaces. The treatment group received an instructional intervention in a classroom while a facilitator in another classroom monitored the control group.

The treatment group studied a more challenging CT concept and activities such as 'variables' and 'creating a quiz using Scratch' with an instructor's guidance and facilitation. Using the 'variable' concept, they were required to apply it in the Scratch programme. Both groups were asked to construct a quiz program. The control group studied at their own pace, with booklet provided and without much guidance. The control group are encouraged to explore Scratch and learn how to create any game they desired. A facilitator was available throughout the treatment phase to assist them when needed.

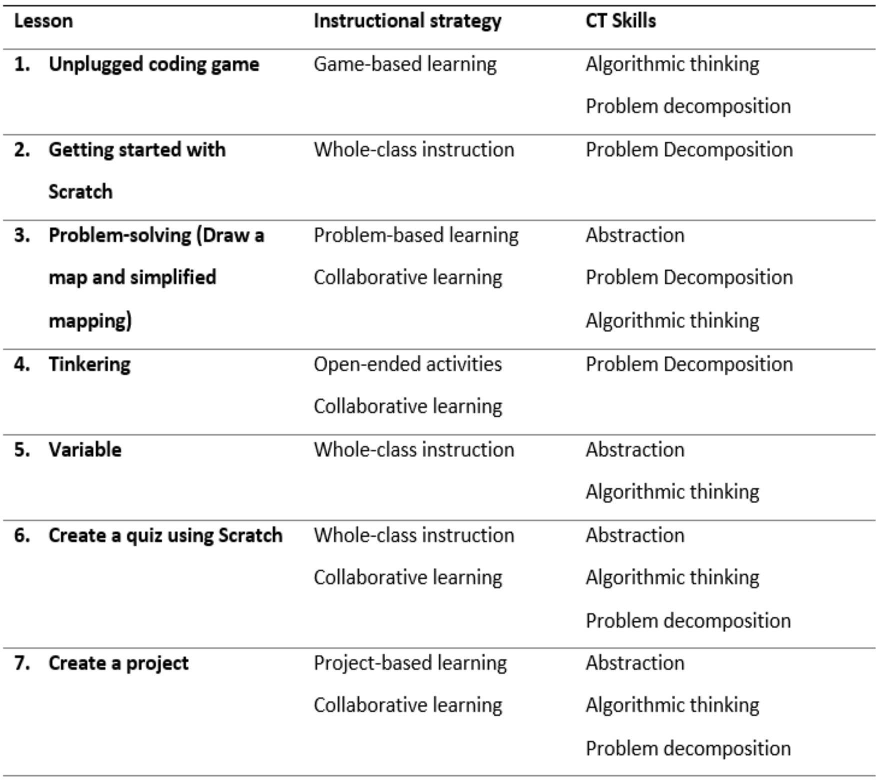
Table 1: Setting for the study experiment



Control Group	Treatment Group
10 students	12 students
4 boys and 6 girls	7 boys and 5 girls
Follow the intervention plan except for lesson 4, 5, and 6. A learner-centred method on the creation of games using the Scratch environment, reference guidebook, and facilitated by an IT teacher.	Follow the complete intervention plan. A mixture of conventional method and learner-centred method assisted by facilitators and a teacher.
The classroom includes 3 small computers and teacher's computer.	The classroom includes 6 small computers, a instructor's computer and LCD projector.

We designed and deployed a set of localised instructional strategies to introduce computational thinking skills such as algorithmic thinking, problem decomposition, and abstraction (see Table 2).

Table 2: Computational Thinking activities and instructional strategies



Lesson	Instructional strategy	CT Skills
1. Unplugged coding game	Game-based learning	Algorithmic thinking Problem decomposition
2. Getting started with Scratch	Whole-class instruction	Problem Decomposition
3. Problem-solving (Draw a map and simplified mapping)	Problem-based learning Collaborative learning	Abstraction Problem Decomposition Algorithmic thinking
4. Tinkering	Open-ended activities Collaborative learning	Problem Decomposition
5. Variable	Whole-class instruction	Abstraction Algorithmic thinking
6. Create a quiz using Scratch	Whole-class instruction Collaborative learning	Abstraction Algorithmic thinking Problem decomposition
7. Create a project	Project-based learning Collaborative learning	Abstraction Algorithmic thinking Problem decomposition

We conducted the lessons on Computational thinking with both groups, and we observed their interactions, behaviours and comprehension while on task. Comprehension was measured through a pre-test and a post-test instrument, which assessed their level of knowledge about Computational thinking. The instrument consists of questions on algorithmic skill, abstraction, and problem decomposition. There is a total of five questions in the instrument; four items focused on knowledge about Scratch™ and Programming, and another item was written to look at algorithmic thinking (see Appendix). Participants were provided thirty minutes to undertake each test. The items were rated using scales ranging from 1 to 4 (from "beginner – showing little understanding of the concept", to "advanced - fluent understanding of the concept".

Photos and videos taken during the lessons were analysed using content analysis. Narrative analysis was used to examine the interactions which took place during the lessons.

4. Results and Findings

Twenty-one responses are collected from both treatment and control groups. One missing data is detected when one participant from the control group was absent during the post-test (see Fig 1). The horizontal axis indicates an ascending order of participants from 1 to 10 while the vertical axis shows the percentage of CT score gained by each sample.

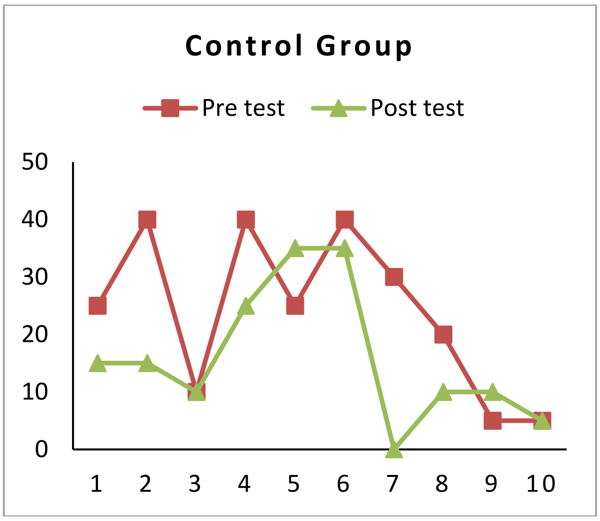


Figure 1: CT results of the Control group

The CT pre-test results show that the treatment group performed better than the control group, with a difference of 3.5% (see Table 3). It shows that participants in the treatment group performed slightly better as compared to the control group.

Table 3: The mean score and differences of scores for both groups

	Control Group	Difference	Treatment Group
Mean score for pre-test	24.0%	3.5%	27.5%
Mean score for post-test	17.8%	11.4%	29.2%
Difference of post- and pre-test	- 6.2%	7.9%	1.7%

After the post-test, the treatment group shows an improvement of 1.7% while the control group shows a decrease of 6.2%. It indicates that whole-class instruction and open-ended activities were helpful for these participants to learn CT skills, even though the results are not conclusive. However, other factors, such as personal language competencies and problem-solving ability, might have also influenced the results. Based on Fig 2, the results of the treatment group show only six participants had improved their post-test performance while others either showed a decrease or their scores remained unchanged.

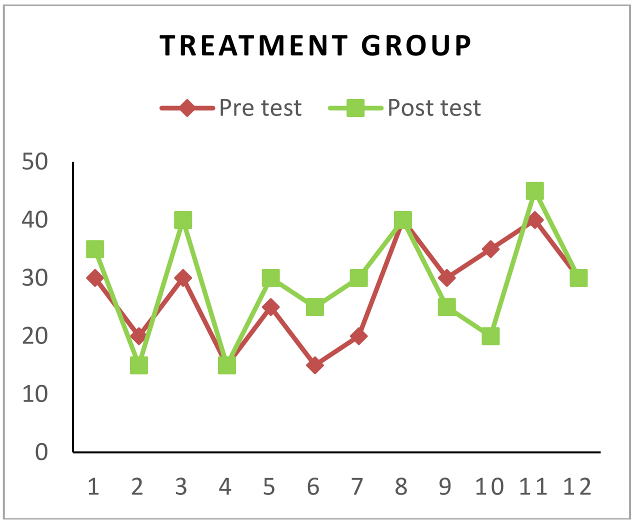


Figure 2: CT results of the Treatment group

Overall, participants demonstrated high motivation during all of the computational thinking activities. In the unplugged game-based coding activity, 22 participants were divided into five groups and these groups are further divided into subgroups. Each group formed two more subgroups of two to three people, as they were instructed to compete with the other subgroups during the game-based activity.

The unplugged coding game produced a mixed reaction among the participants. Although they are repeatedly reminded of the game mechanics, most still failed to comprehend them. Some of the female participants found it challenging to comprehend the mechanics of the game, and they eventually gave up playing. Based on our observations, the game might not have resembled anything they have played before, and they might have lost interest in it because the game did not present a worthy challenge to them. The phenomenon was evident among those who did not notably score high in the pre-test and post-test. We assumed that their performance was slowed by the unplugged coding game, which had required participants to use abstract thinking skills to identify steps and directions to the desired goal (see Fig 3). In our observation, it was a challenge for them to think abstractly and systematically. We conclude that these participants might have required extended support and repetitive scaffolding to guide them to internalise the "foreign" algorithmic concepts.

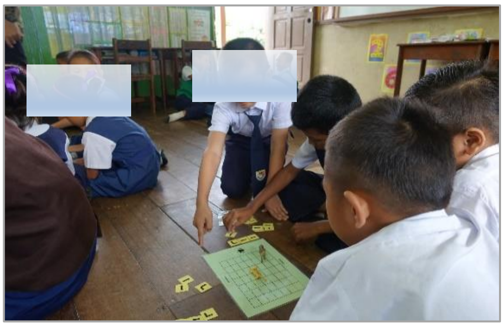


Figure 3: Participants attempting to play an unplugged algorithmic thinking game

In the "Getting started with Scratch" activity, the Scratch™ programme was introduced to the participants to get them to familiarise with the blocks and utilise the tool. The Malay language is used as a medium to teach them Scratch. Notes are given to guide them getting started with Scratch in case they missed the steps demonstrated by the instructor. It was observed that during the sessions, the participants communicated in their mother tongue, the Penan language. The code-switching between Penan and Malay as they interacted with the facilitators and among themselves seemed to have supported their understanding of the assigned tasks.

In the problem-solving session, participants were assigned to draw a map based on a scenario given and label the vital information such as location, places (schools), object (boat), and time of departure and arrival (see Fig 4). This lesson aims to practice abstraction skill, decomposition skill, and algorithmic thinking. Before drawing the map, they were briefed with a familiar situation of a travelling problem. The facilitator used a similar example and demonstrated an illustration for them to visualise the intended situation. The details of information were added to the map as essential elements for mark-up and problem-solving.

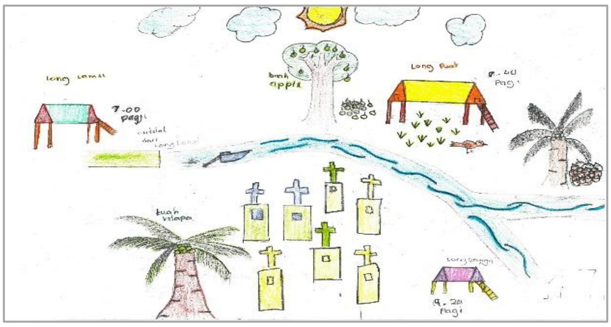


Figure 4: A travel map was drawn by using Abstraction skills

During the tinkering session, the participants began to use Scratch™ to complete the tasks given. Most of the groups managed to complete the tasks. Some of the participants have had to seek further guidance and support, as they were uncertain about the steps to add or change the sprites in Scratch™.

To measure participant's understanding of using Scratch™ as a tool for programming, questions such as "Do you know what the sprite was doing in the program?" and "Could you describe the flow of the blocks you generated just now?" were asked to the participants. A participant gave only a partial answer as he could not understand the flow of the Scratch™ programme. Our observations revealed that some participants had needed more time and support to fully understand how Scratch™ works. Some had trouble understanding the value of basic steps when arranging code blocks in the programme. However, throughout all of the sessions, they displayed a high level of curiosity and determination. They intentionally made an effort to try parameters to test and play the sprites within Scratch, a sign of comprehension progress.

At times, in between the sessions, we observed that some of the female participants showed anxiety and displacement. The male participants generally showed a higher interest in learning programming. When they were able to use the code blocks successfully to create new games, they displayed enthusiasm and excitement. The male participants overall presented a more serious commitment to the assigned tasks. When they found a problem, they would take the initiative to send someone from their team to ask for clarification from the facilitator.

In another instance, one group successfully created a new sprite using a painting tool in Scratch. It was not taught in any of the sessions. While creating the sprite's design, one female participant from the group faced issues navigating with the mouse. Her limited navigational skills hampered her effort to complete the design of the new sprite for her team. It was interesting to observe the group interaction and support transpired throughout the predicament faced by the leader in their team, much of which contributed to the success of the group as a whole.

Before any of the session started, the participants were divided into two different classes. Twelve participants are grouped as the treatment group in a classroom while the rest in the control group in another classroom. We only fully observed the treatment group while the control group was left on their own to manage the assigned tasks.

In the first session, the twelve participants in the treatment group were divided into four groups. A facilitator started to introduce the concept of variables to them and demonstrating on the whiteboard. All participants paid close attention throughout the explanation. Daily examples local to the participants were used to elaborate on the concepts introduced in the session. They learned to make connections between variables made up of their own names, favourite subjects in school and the daily items they used at home. In the following session, the participants learned how to create a quiz using Scratch™. Using the variable concept they had just learned, they learned a numeric example and experimented on it to create a conditional statement (IF-THEN).

```
IF number > 1,
THEN True,
ELSE False.
```

Using examples which have already been discussed, the participants learned to apply the information into a quiz in Scratch™. Then, the participants construct their arithmetic equation and write it on paper. All participants were able to complete the tasks without difficulties.

In these initial sessions, both treatment and control groups were placed in the same classroom. Before the session began, the participants were given options to choose what type of project they want to create. Majority of them chose to create Scratch game, and only one group chose to create a Math quiz. During the project creation phase, the participants used decomposition skills to arrange the blocks and designed the structures of their team's programmes. They selected each block carefully and connected it to the appropriate block to create a smooth flow of the intended action. Most of the groups who had opted to create a Scratch™ game were observed to have preferred to read a provided guidebook which contained all information about creating games in Scratch™.

The Math quiz group, surprisingly, worked very well with each other and were on task. They were able to complete the Math quiz programme collaboratively and had asked for minimal assistance throughout the creation process. They invited the facilitators to try out their Math quiz as soon as they were done. The group showed excitement, relief and pride when their programme tested well. At this point, the group members began to explore Scratch™ even further. They started experimenting with creating new sprites on their own. We observed that they were able to cope with the task by referring to the guidebook provided.

Participants then used decomposition skills during the project development to arrange the code blocks and design the program. Each block should be selected carefully and attached to the corresponding block to ensure a smooth output. They began to familiarise themselves with the use of a mouse and create a Scratch game without the help of the teacher and facilitator.

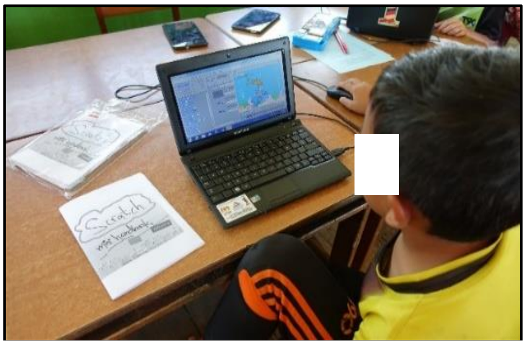
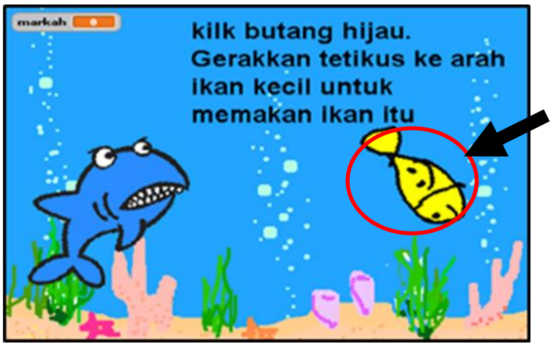


Figure 5: A participant created a game in Scratch using Decomposition skills

Fig 6 shows a new Fish sprite drawn by the participants without being instructed. Their unsupervised attempts demonstrated their ability and keenness to work independently and use available materials such as Scratch booklet to create new sprites.



kilk butang hijau.
Gerakkan tetikus ke arah
ikan kecil untuk
memakan ikan itu

New sprite
is drawn by
one of the
participants.

Figure 6: A fish game built by one of the groups

In summary, these participants able to adopt computation thinking skills such as abstraction, problem decomposition, and algorithmic thinking. After a few sessions, the participants able to familiarise with the use of Scratch and apply CT skills. Although each group created little during the task, they showed positive engagement during the activities such as asking questions, helping others and play with their peers. In contrast, these activities were not able to determine their level of CT skills because the instructor and facilitators guided all activities.

During collaborative learning, the participants worked on an art-based activity, exploration with Scratch™, and creating a Scratch™ project. We saw how most of the participants enjoyed drawing and colouring activity, notably much more than the programming tasks. Brennan and Resnick (2012) described that not only pointing, clicking, browsing, and chatting as skills to develop a computational thinker. A computational thinker can do more than a consumer, whereby they use computation skills for computational expressions (Brennan and Resnick, 2012).

Although the students had never experienced working with numbers before they participated in the study, they were able to cope with the computational thinking tasks. Observations captured in the study were synonymous with Yadav et al. 's (2014) suggestion on young children being introduced early to computational thinking skills to solve computational problems. Scratch™ is a powerful tool for children who are inexperienced to practice CT and think like a designer. Throughout their use of computational thinking skills, these children were also becoming more interested in using technology for problem-solving.

Figure 7: A participant drew a new sprite using the Editor tool in Scratch

As observed, many children enjoyed participating in art-based activity (see Fig 7), which is comfortable and freer to do what they want. The study suggests that children can be creative and innovative if they are encouraged to explore the topic through art and active learning. Furthermore, they are more likely to recall their learning experience through meaningful experience, which linked to creative problem-solving and discovery learning (Winner et al., 1888). The same authors also found that art education could enhance students' learning motivation, reducing the number of students in absenteeism, and promoting greater participation. At the end of the activity, these children can perceive technology as a powerful medium to express creative ideas rather than solely following the teacher's instruction. When doing project-based learning, they appeared to have asked more questions as they were eager to know about which function blocks were required to make the sprites to move.

5. Conclusion

Based on CT performance during the study, it showed that these novice indigenous children were able to learn and use the "foreign" concepts of abstraction, decomposition, and algorithmic thinking. They exhibited learning characteristics such as 'learning-by-making', collaborative, highly motivated, playfulness, curiosity, and creativity. Although the CT post-test results showed that understanding of computational concepts was poor, confidence and engagement levels have visibly increased during the construction of their programming projects. We observed their need for supportive peers, guided scaffolding to learn the contents, and a positive learning environment. Language proficiency also played a role in enabling the learning of "foreign" concepts such as computational thinking in a remote setting as such. The codeswitching from Penan to Malay in verbal interactions, and to English as they read prompts from Scratch, might have affected the way the participants understood computational thinking skills.

The study has illustrated the effectiveness of mixing game-based learning, collaborative learning, problem-based learning, and project-based learning to present a new "foreign" concept to young novice indigenous children in engaging and understanding considered CT skills. These techniques, when used with localised examples which exist in the participants' home environment, were deemed culturally adaptable to the learning of computational thinking skills.

Preliminary results from the control group's post-test suggest that an unguided instructional approach is less effective than guided instructional strategies. Future research is needed to pursue a deeper understanding of the development of CT skills among young novice learners such as these Penan school children. Computational thinking skills may not be an obvious schema in their living environment and teaching such concepts would require a deeper understanding of how teachers could make connections between indigenous knowledge and curriculum-prescribed learning contents.

6. Acknowledgement

The research project was funded by Newton AHRC-MoHE (UK SEA-NUOF JPTS (BPKI)2000/04/07/05(33). The authors appreciate the support of the Institute Social Informatics and Technological Innovations (ISITI) and Dr John Phua from Universiti Malaysia Sarawak (UNIMAS), the Ministry of Education, Sarawak State Education Department, and guidance from all teachers at SK Long Lamei throughout the study.

7. References

Asad, K., Tibi, M., & Raiyn, J. (2016). Primary School Students' Attitudes toward Learning Programming through Visual Interactive Environments. *World Journal of Education*, *6*(5), 20–26. https://doi.org/10.5430/wje.v6n5p20

Baharom, S., & Palaniandy, B. (2013). Problem-Based Learning: A Process for the Acquisition of Learning and Generic Skills. *The 4th International Research*

Symposium on Problem-Based Learning (IRSPBL), 47–55. Retrieved from http://tree.utm.my/wp-content/uploads/2013/11/Problem-Based-Learning-A-Process-for-the-Acquisition-of-Learning-and-Generic-Skills.pdf

Berger, D. N. (2019). *The Indigenous World 2019*. Retrieved from https://www.iwgia.org/en/documents-and-publications/documents/4-the-indigenous-world-2019/file.html

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palinscar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist*, *26*(3–4), 369–398.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *AERA*, 1–25. Retrieved from https://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf

Cantillon, P., Hutchinson, L., & Wood, D. (2003). *ABC of learning and teaching in medicine: Learning and teaching in the clinical environment*. London, England: BMJ Publishing Group Ltd. Retrieved from http://edc.tbzmed.ac.ir/uploads/39/CMS/user/file/56/scholarship/ABC-LTM.pdf

Creswell, J. W. (2012). *Educational research planning conducting and evaluating quantitative and qualitative research* (4th ed.). Boston, MA: Pearson.

Creswell, J. W., & Piano Clark, V. L. (2007). Designing and conducting mixed methods research. *Australian and New Zealand Journal of Public Health*, *31*(4), 388–388. https://doi.org/10.1111/j.1753-6405.2007.00096.x

Durkin, S. S., Fees, R. E., da Rosa, J. A., Murray, M. M., & Moran, A. L. (2018). Unplugged cybersecurity: An approach for bringing computer science into the classroom. *International Journal of Computer Science Education in Schools*, *2*(1). https://doi.org/10.21585/ijcses.v2i1.21

Grover, S. (2018). *The 5th 'C' of 21st Century Skills ? Try Computational Thinking (Not Coding)*. Retrieved from https://www.edsurge.com/news/2018-02-25-the-5th-c-of-21st-century-skills-try-computational-thinking-not-coding

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Haseski, H. I., Ilic, U., & Tugtekin, U. (2018). Defining a new 21st Century Skill-Computational Thinking: Concepts and trends. *International Education Studies*, *11*(4), 29. https://doi.org/10.5539/ies.v11n4p29

Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers and Education*, *126*(July), 296–310. https://doi.org/10.1016/j.compedu.2018.07.004

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers and Education*, 82(March), 263–279. https://doi.org/10.1016/j.compedu.2014.11.022

Kafai, Y. B., & Burke, Q. (2014). Connected gaming: Towards integrating instructionist and constructionist approaches in K-12 serious gaming. *Proceedings of International Conference of the Learning Sciences, ICLS*, *1*(January), 86–93. Retrieved from https://www.researchgate.net/publication/287602204_Connected_gaming_Towards_integrating_instructionist_and_constructionist_approaches_in_K-12_serious_gaming

Kafai Y.B., Burke Q. (2017). Computational participation: Teaching kids to create and connect through code. In Rich P., Hodges C. (eds), *Emerging Research, Practice, and Policy on Computational Thinking. Educational Communications and Technology: Issues and Innovations (ECTII)*. Springer, Cham. https://doi.org/10.1007/978-3-319-52691-1_24

Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning programming at the computational thinking level via digital game-play. *Procedia Computer Science*, *9*(December), 522–531. https://doi.org/10.1016/j.procs.2012.04.056

Kementerian Pendidikan Malaysia [KPM]. (2014). *RPT TAHUN 1 KSSR RPT Dunia Sains Dan Teknologi (SK) Tahun 1*. Retrieved from http://www.sistemguruonline.my/2014/10/rpt-tahun-1-kssr-rpt-dunia-sains-dan.html

Law, S. (2018) Vitaliy of Mother Tongue Language Among Penan and Chinese Communities in Sarawak. Universiti Malaysia Sarawak. Unpublished thesis.

Lin, V. J. (2015). *Computational thinking and technology toys* (Honors Thesis Collection, Wellesley College). Retrieved from https://repository.wellesley.edu/thesiscollection/307

Ling, U. L., Saibin, T. C., Labidin, J., & Aziz, N. A. (2017). Preliminary investigation: Teachers' perception of computational thinking concepts. *Journal of Telecommunication, Electronic and Computer Engineering*, *9*(2), 23–29. Retrieved from https://www.researchgate.net/publication/324886429_Preliminary_Investigation_Teachers'_Perception_on_Computational_Thinking_Concepts

Malaysia Digital Economy Corporation [MDEC]. (2019). *Who we are*. Retrieved from https://mdec.my/about-mdec/who-we-are/

Ministry of Education [MOE]. (2013). Malaysia Education Blueprint 2013-2025 (Preschool to Post-Secondary Education). *Kementerian Pendidikan Malaysia*. Retrieved from https://www.moe.gov.my/menumedia/media-cetak/penerbitan/dasar/1207-malaysia-education-blueprint-2013-2025/file

Mohamad, F. S., Yeo, A. W., Abd Aziz, N., & Soubakeavathi, R. (2010). Borneo children in an international digital playground: Intercultural issues and idiosyncrasies. In *Proceedings of the 3rd ACM International Conference on Intercultural Collaboration (ICIC '10*). Association for Computing Machinery, New York, USA. https://doi.org/10.1145/1841853.1841869

Moschella, M. (2019). Observable computational thinking skills in primary school children: How and when teachers can discern abstraction, decomposition and use of algorithms. *INTED2019 Proceedings*, *1*(March), 6259–6267. https://doi.org/10.21125/inted.2019.1523

Mustafa, Z. (2019, March 20). STEM policies set for an evolution. *New Straits Times*. Retrieved from https://www.nst.com.my/education/2019/03/471176/stem-policies-set-evolution

Paniagua, A., & Istance, D. (2018). *Teachers as designers of learning environments: The importance of innovative pedagogies* [e-book]. Educational Research and Innovation, OECD Publishing, Paris. https://doi.org/10.1787/9789264085374-en

Papert, S., & Harel, I. (1991). Situating Constructionism. *Constructionism*, 1–17. Retrieved from http://web.media.mit.edu/~calla/web_comunidad/Reading-En/situating_constructionism.pdf

Perković, L., Settle, A., Hwang, S., & Jones, J. (2010). A framework for computational thinking across the curriculum. *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education - ITiCSE '10*. https://doi.org/10.1145/1822090.1822126

Robles, G., Barone, D., Brackmann, C. P., Román-González, M., Moreno-León, J., & Casali, A. (2017). Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education (WiPSCE '17), ACM, November*, 65–72. https://doi.org/10.1145/3137065.3137069

Selby, C. C. (2014). *How can the teaching of programming be used to enhance computational thinking skills?* (Doctoral dissertation, University of Southampton, England, United Kingdom). https://doi.org/10.1016/j.jsv.2010.04.020

Tseng, K. H., Chang, C. C., Lou, S. J., & Chen, W. P. (2013). Attitudes towards science, technology, engineering and mathematics (STEM) in a project-based learning (PjBL) environment. *International Journal of Technology and Design Education*, *23*(1), 87–102. https://doi.org/10.1007/s10798-011-9160-x

Unnikrishnan, R., Amrita, N., Muir, A., & Rao, B. (2016). Of elephants and nested loops: How to introduce computing to youth in rural India. In *Proceedings of IDC 2016 - The 15th International Conference on Interaction Design and Children*, 137–146. https://doi.org/10.1145/2930674.2930678

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*. https://doi.org/10.1098/rsta.2008.0118

Wing, J. M. (2010). Computational thinking: What and why?. *The link - The Magazine of the Carnegie Mellon University School of Computer Science*. Retrieved from https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Winner, E., Goldstein, T. R., & Vincent-Lancrin-Stéphan. (1888). Art for art's sake?: Overview. *OECD*. Retrieved from http://www.oecd.org/education/ceri/ART%20FOR%20ART%E2%80%99S%20SAKE%20OVERVIEW_EN_R3.pdf

Xie, Q. (2016). Research design and methods. In English Language Training in the Workplace. *Technical and Vocational Education and Training: Issues, Concerns and prospects*, *22*, 99–116. Springer, Cham. https://doi.org/10.1007/978-3-319-30157-0_5

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st-century problem-solving in K-12 classrooms. *TechTrends*, *60*(6), 565–568. https://doi.org/10.1007/s11528-016-0087-7

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, *14*(1). https://doi.org/10.1145/2576872

Zaman, T., & Falak, H. (2019). Framing indigenous knowledge in the digital context: Technologies, methods and tools. *International Journal of End-User Computing and Development*. https://doi.org/10.4018/ijeucd.2018070103

Zaman, T., Kulathuramaiyer, N., & Yeo, A. W. (2016). eToro: Appropriating ICTs for the management of Penans' indigenous botanical knowledge. In L. E. Dyson, L. E., S. Grant, & M. Hendriks (eds.), *Indigenous people and mobile technologies: Routledge studies in new media and cyberculture* (1st ed., pp. 253-264). New York: Routledge. https://doi.org/10.4324/97813157593

Appendix 1

Computational Thinking Questions

Soalan Pemikiran Komputational

(1)

Nama: ______________________________

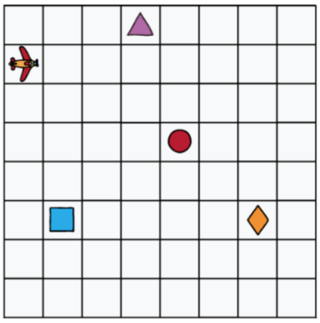
Jantina: Lelaki / Perempuan

Kelas Darjah: __________

2018

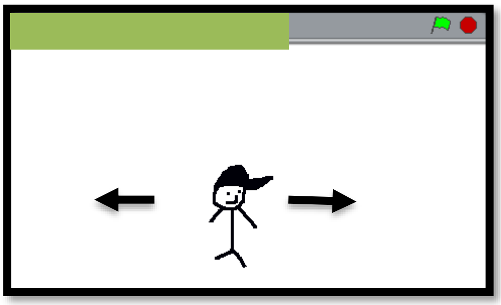
Computational thinking pre-test questions

1. The plane needs to find its way to the circle. Write the steps on how to move to the destination. You may use instructions such as go forward, turn right, and turn left.
   Pesawat perlu mencari laluannya ke bulatan. Tulis langkah-langkah bagaimana untuk bergerak ke destinasi. Anda boleh menggunakan set arahan ke hadapan, belok kanan, dan belok kiri.



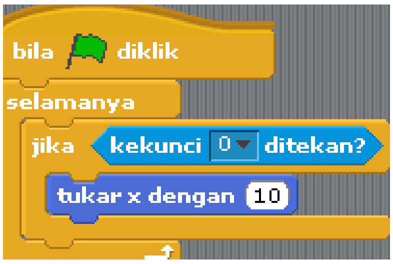
Jawapan

2. The sprite is going forward when you press the '0' button. But it cannot move backwards. Could you help to solve this problem? Write your solution at any space below.
Sprit akan bergerak ke hadapan apabila anda menekan butang '0'. Tetapi ia tidak dapat bergerak ke belakang. Bolehkah anda membantu menyelesaikannya? Tulis penyelesaian anda di mana-mana ruang di bawah.



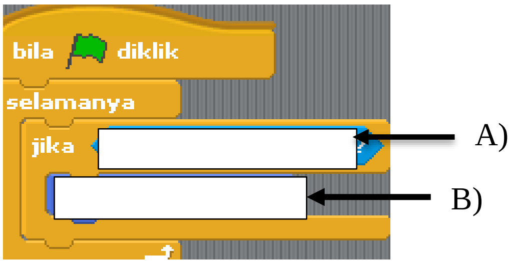
Jawapan:

<u>Bergerak ke hadapan</u>



```
bila 🏁 diklik
selamanya
  jika kekunci 0 ditekan?
    tukar x dengan 10
```

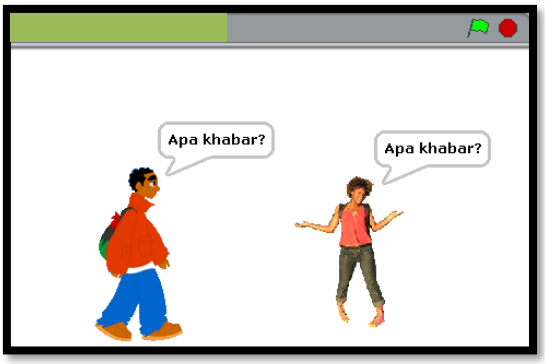
<u>Bergerak ke belakang</u>

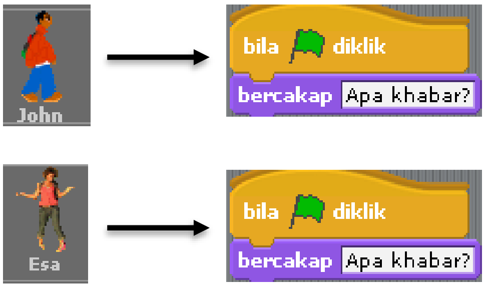


```
bila 🏁 diklik
selamanya
  jika
```

3. Your class teacher has assigned you a task which is to create a program. There are two characters, John and Esa in the program. The condition is John will say “Apa khabar?” to Esa and Esa will say the same thing to John. But they say it at the same time. How to fix that?

   Guru kelas anda telah memberikan anda satu tugas iaitu menghasilkan satu program. Terdapat dua watak, John dan Esa dalam program ini.Syaratnya ialah John akan berkata "Apa khabar?" kepada Esa dan Esa akan mengatakan perkara yang sama kepada John. Tetapi John dan Esa mengatakannya pada masa yang sama. Bagaimana untuk membetulkannya?

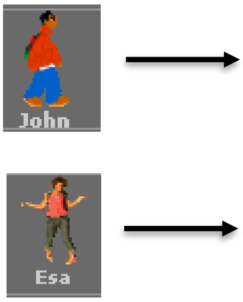




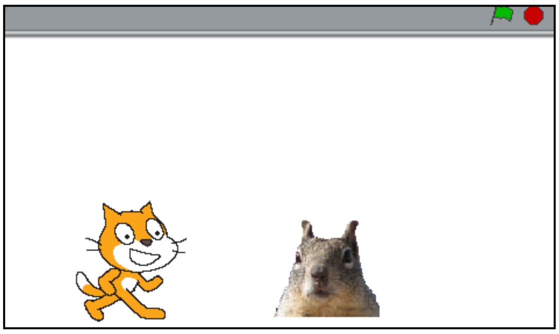
```
John →
bila 🏁 diklik
bercakap Apa khabar?

Esa →
bila 🏁 diklik
bercakap Apa khabar?
```

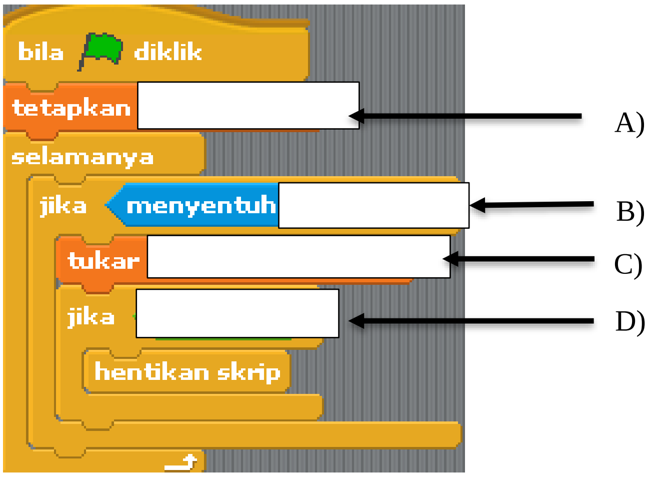
Jawapan



4. A cat has nine lives. If it collides with a squirrel, one life is lost. If all nine lives are lost, the game is over. Fill in the blanks in this script to make these rules work.
Seekor kucing mempunyai sembilan nyawa. Jika ia bertembung dengan tupai, satu nyawa akan hilang. Sekiranya semua nyawa hilang, permainan ini akan ditamatkan. Isi ruang kosong dalam skrip ini untuk membuat peraturan ini berfungsi.

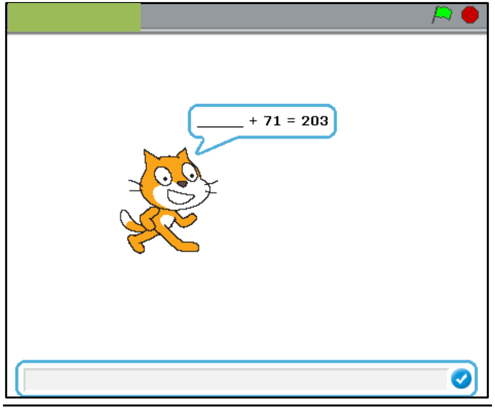


Jawapan



5. Your class teacher has assigned you to create a calculator machine by programming. The problem is "What number to add with 71 so that you get 203?"

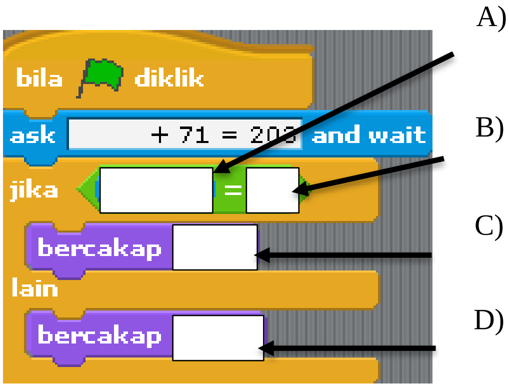
Guru kelas anda telah menetapkan anda untuk membuat matematik kalkulator dengan pengaturcaraan. Masalahnya ialah "Nombor apakah yang perlu ditambah dengan 71 untuk mendapatkan 203?"



Definition (Istilah):

- Ask (bertanya)
- Wait (tunggu)

<u>Jawapan</u>



```
bila [flag] diklik
ask [ + 71 = 203] and wait
jika < [ ] = [ ] >
    bercakap [ ]
lain
    bercakap [ ]
```

Appendix 2

Computational Thinking Questions

Soalan Pemikiran Komputational

(2)

Nama: ______________________________

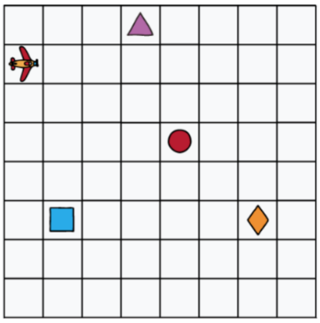
Jantina: Lelaki / Perempuan

Kelas Darjah: __________

2018

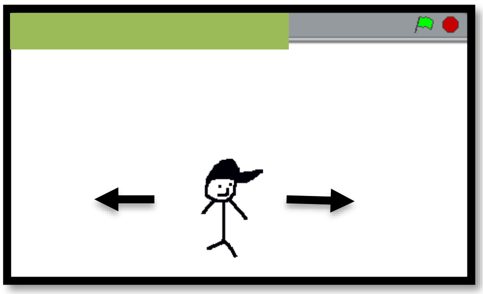
Computational thinking post-test questions

1. The plane needs to find its way to the circle. Write the steps on how to move to the destination. You may use instructions such as go forward, turn right, and turn left.
Pesawat perlu mencari laluannya ke bulatan. Tulis langkah-langkah bagaimana untuk bergerak ke destinasi. Anda boleh menggunakan set arahan ke hadapan, belok kanan, dan belok kiri.



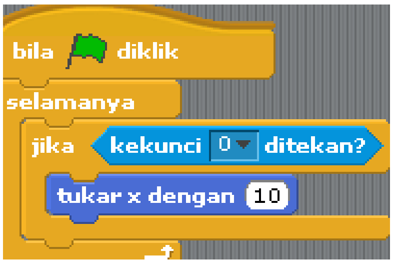
Jawapan

2. The sprite is going forward when you press the ‘0’ button. But it cannot move backwards. Could you help to solve this problem? Write your solution at any space below.
Sprit akan bergerak ke hadapan apabila anda menekan butang '0'. Tetapi ia tidak dapat bergerak ke belakang. Bolehkah anda membantu menyelesaikannya? Tulis penyelesaian anda di mana-mana ruang di bawah.



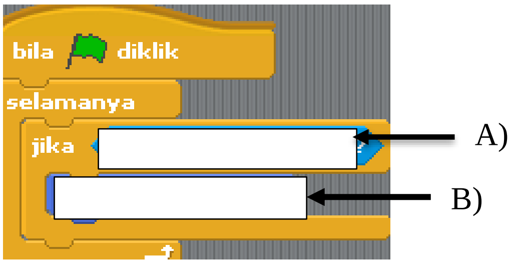
Jawapan:

<u>Bergerak ke hadapan</u>



```
bila ⚑ diklik
selamanya
  jika kekunci 0 ditekan?
    tukar x dengan 10
```

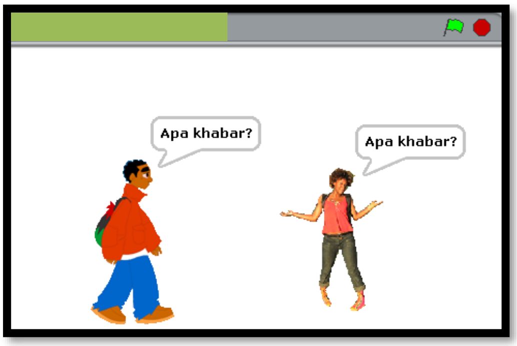
<u>Bergerak ke belakang</u>

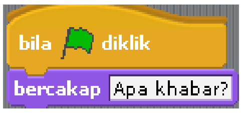


```
bila ⚑ diklik
selamanya
  jika
```

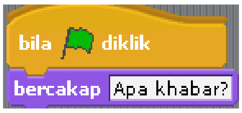
3. Your class teacher has assigned you a task which is to create a program. There are two characters, John and Esa in the program. The condition is John will say "Apa khabar?" to Esa and Esa will say the same thing to John. But they say it at the same time. How to fix that?

   Guru kelas anda telah memberikan anda satu tugas iaitu menghasilkan satu program. Terdapat dua watak, John dan Esa dalam program ini.Syaratnya ialah John akan berkata "Apa khabar?" kepada Esa dan Esa akan mengatakan perkara yang sama kepada John. Tetapi John dan Esa mengatakannya pada masa yang sama. Bagaimana untuk membetulkannya?



→


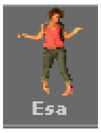
```
bila 🏁 diklik
bercakap Apa khabar?
```

→


```
bila 🏁 diklik
bercakap Apa khabar?
```

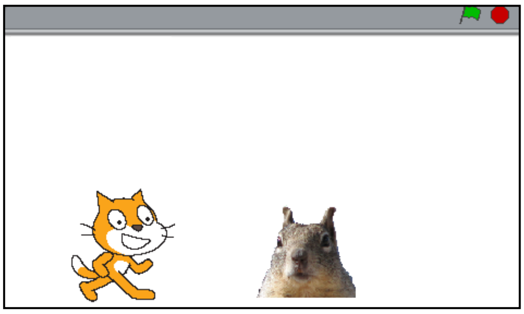
Jawapan

→

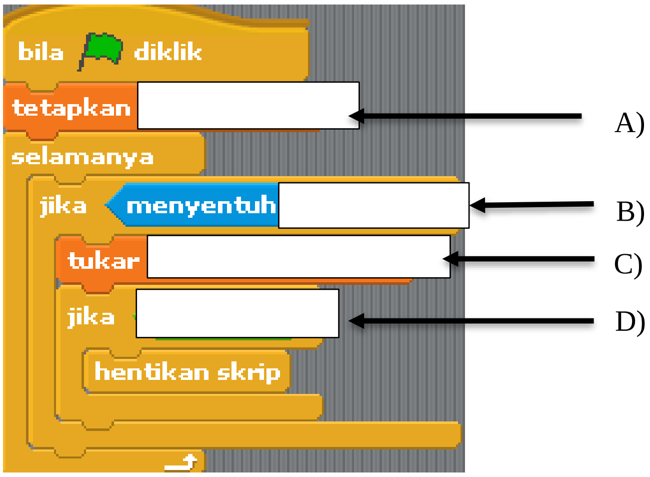


→

4. A cat has eleven lives. If it collides with a squirrel, one life is lost. If all eleven lives are lost, the game is over. Fill in the blanks in this script to make these rules work.
Seekor kucing mempunyai sebelas nyawa. Jika ia bertembung dengan tupai, satu nyawa akan hilang. Sekiranya semua nyawa hilang, permainan ini akan ditamatkan. Isi ruang kosong dalam skrip ini untuk membuat peraturan ini berfungsi.



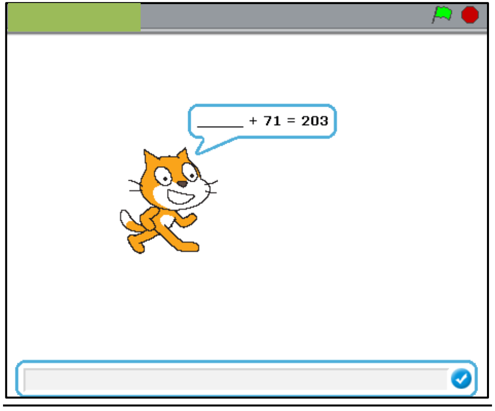
Jawapan



```
bila [bendera] diklik
tetapkan [          ]
selamanya
  jika menyentuh [          ]
    tukar [          ]
    jika [          ]
      hentikan skrip
```

5. Your class teacher has assigned you to create a calculator machine by programming. The problem is "What number to add with 71 so that you get 203?"

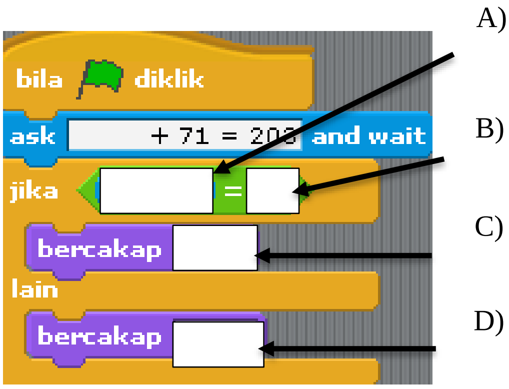
Guru kelas anda telah menetapkan anda untuk membuat matematik kalkulator dengan pengaturcaraan. Masalahnya ialah "Nombor apakah yang perlu ditambah dengan 71 untuk mendapatkan 203?"



Definition (Istilah):

- Ask (bertanya)
- Wait (tunggu)

Jawapan


A)
bila
diklik
ask
+ 71 = 203
and wait
B)
jika
=
C)
bercakap
lain
D)
bercakap