IMPROVING CHILDREN'S COMPUTATIONAL THINKING SKILLS THROUGH TANGIBLE TECHNOLOGY

TITIPAN PHETSRIKRAN

A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Master of Science in (Computer Science)
2020

IMPROVING CHILDREN'S COMPUTATIONAL THINKING SKILLS THROUGH TANGIBLE TECHNOLOGY

TITIPAN PHETSRIKRAN

A Thesis Submitted to the Graduate School of Naresuan University
in Partial Fulfillment of the Requirements
for the Master of Science in (Computer Science)
2020

Thesis entitled "Improving Children's Computational Thinking Skills Through Tangible Technology"

By TITIPAN PHETSRIKRAN

has been approved by the Graduate School as partial fulfillment of the requirements for the Master of Science in Computer Science of Naresuan University

Oral Defense Committee

.... Chair
(Assistant Professor Attawith Sudsang, Ph.D.)

.... Advisor
(Wansuree Masagram, Ph.D.)

.... Co Advisor
(Supiya Charoensiriwath, Ph.D.)

.... Internal Examiner
(Antony James harfield, Ph.D.)

.... Internal Examiner
(Assistant Professor Thanathorn Phoka, Ph.D.)

Approved

....
(Professor Paisarn Muneesawang, Ph.D.)

Dean of the Graduate School

Title IMPROVING CHILDREN’S COMPUTATIONAL THINKING SKILLS THROUGH TANGIBLE TECHNOLOGY
Author TITIPAN PHETSRIKRAN
Advisor Wansuree Masagram, Ph.D.
Co-Advisor Supiya Charoensiriwath, Ph.D.
Academic Paper Thesis M.S. in Computer Science, Naresuan University, 2020
Keywords Computational Thinking, Educational Robotics, Tangible Technology, Mobile Technology

ABSTRACT

The purpose of this study is to design, implement, and test the feasibility of an educational platform for primary students' computational thinking learning. The Arducation Bot combines tangible technology and mobile technology to create intuitively approachable teaching computational thinking. The Arducation Bot system was tested with 177 primary students from Phitsanulok Thailand. A clear pattern of improved computational thinking was demonstrated by the pre-test and post-test scores and related data from the Arducation Bot. This low-cost and intuitive teaching tool can potentially develop skills in computational thinking and prepare students for computer science.

ACKNOWLEDGEMENTS

This study was made possible by a generous grant from the Thailand Graduate Institute of Science and Technology (TGIST) fund (grant no. SCA-CO-2560-4530-TH). The authors would also like to thank the NU Maker Club and the Mobile Computing Laboratory at Naresuan University for their assistance.

TITIPAN PHETSRIKRAN

TABLE OF CONTENTS

Page

List of Tables

Page

List of Figures

Page

Chapter 1
Introduction

1.1 Introduction

The term “computational thinking” refers to a thought process to develop problem-solving skills that can be taught and learned. This method of thinking breaks down any task into smaller parts, finding patterns in each problem, and then logically presenting solutions using algorithms that a machine could follow.

Computational thinking is what a software developer uses to develop programs. Not only computational thinking helps solve problems in computer science and mathematics but also in everyday life. Given its current relevance and importance, there is much demand in Thailand for teaching computational thinking in schools starting at a young age (Reeve, 2013). One focus of the Thai government’s current “Thailand 4.0” policy is to foster a “Learning Society”, which involves moving away from a learning style that is standardized and based on duty in favor of a learning style that is personalized and based on passion. Thailand's current national education plan (2017-2036) designates the following as skills required for the 21st century: critical thinking, problem-solving, cross-cultural communication, collaboration, leadership, communication, information and media literacy, computing, ICT literacy, career learning, and compassion (Council., 2017).

Recognizing the importance of computational thinking as a critical component for achieving these essential skills, the Ministry of Education has promoted the teaching of computational thinking at an early age. To that end, Thailand has introduced computational science as a compulsory subject in both the primary and secondary curricula of the country’s school system. The Institute for the Promotion of Teaching Science and Technology (IPST) has created a new curriculum called "Computational Science" for Thai students in primary and secondary schools. The National Science and Technology Development Agency (NSTDA), another Thai government agency, has developed a hardware system called "Kid Bright" (NECTEC, 2018) as a tool for students to learn to code at school. Clearly interest in teaching computational thinking has steadily increased within the last few years.

The purpose of this work is to provide a way to increase computational thinking skills by combining tangible and mobile technologies to develop a platform that is both accessible and effective. The mobile application in this study is an iOS application that delivers challenging puzzles, which are divided into four units, with each unit teaching an important concept of computational thinking: sequences, loops, conditions, and conditions with loops. The tangible technology here is an Arduino-based robot car that young students can hold in their hands and play within an experimenting way. To date there have been two initial versions (T Phetsrikran, Massagram, & Harfield, 2017; Titiphan Phetsrikran, Massagram, Phoka, & Harfield, 2018) of Arducation Bot. The purpose of the current study is to further improve the Arducation Bot and its integrated courseware beyond the initial versions (T Phetsrikran, Harfield, Charoensiriwath, & Massagram, 2021). Specifically, this study will overhaul and improve the existing courseware to teach the target computational thinking skills even more effectively, and measure student improvement in a reliable quantitative way.

1.2 Objectives

The purposes of the study are:

- To create an easily accessible platform with comprehensive courseware to teach computational thinking skills via mobile technology and tangible technology
- To evaluate children's computational thinking skills via mobile technology and tangible technology

1.3 Scopes

This study will cover the following:

- Implementing the mobile application and tangible technology platform for teaching computational thinking in children.
- Establishing comprehensive courseware to increase children's computational thinking skills.
- Evaluating the performance of the platform and courseware

1.4 Definitions

1.4.1 Computational Thinking

Computational Thinking (Dek-D., 2017) is a problem-solving process based on logic, sequencing, data analysis, and the step-by-step solutions, as well as digesting problems that help with complex problems. Computational thinking is essential to developing applications for computers. This way of thinking can also help solve problems in various subjects. By integrating computational thinking through courses, children will see the relationship between each subject and be able to apply these skills to real-life problems.

1.4.2 Educational Robotics

Educational robotics (Wikipedia, 2020a) is a teaching of robot design, analysis, application, and operation; and can be taught from primary school to graduate programs. The robot's major function is to stimulate and help the teaching of other fundamental topics, e.g., computer programming, artificial intelligence, or engineering design.

1.4.3 Tangible Technology

Tangible Technology (Wikipedia, 2020f) or "tangible user interface (TUI)" is the user interface where humans interact with digital information via the physical environment. The goal of TUI development is to enhance collaboration, learning, and design by giving physical models to digital information. Thus, taking advantage of the human potentiality to catch and handle physical objects and stuff.

1.5 Benefits

- Understand how to create a learning platform to teach computational thinking.
- Increase knowledge and ability of children's computational thinking skills to mobile and tangible technology.

Chapter 2

Literature Review

This study is divided into three main topics: computational thinking, educational robotics, and courseware. Chapter 2 addresses background and literature review of each topic as follows.

2.1 Computational Thinking

J.M. Wing claims in her study that anyone can benefit from thinking computationally. Computational thinking will be a fundamental skill similar to reading, writing, and arithmetic; and will be used by everyone by the middle of the 21st Century (Jeannette M. Wing, 2014). The definition of computational thinking is a basic process for solving problems. This skill is required for programmers and computer scientists (Jeannette M Wing, 2006). Six key concepts of computational thinking and five approaches (Barefoot, 2014; BBC, 2017) are listed below:

1. Logic is reasoning that helps us explain why something happens.
2. Algorithms are a sequence of instructions to solve problems.
3. Decomposition is a process of breaking down a task into smaller pieces.
4. Patterns are identifying details, creating rules, and solving more general problems.
5. Abstracting is simplifying or identifying something important without worrying about the details.
6. Evaluation is about making estimates of an objective in a systematic way.

These concepts can improve the development of the five approaches which are:

1. Tinkering is often to try something new to discover how it works.
2. Creating is about making and planning something.
3. Debugging is finding and fixing errors in code or algorithms.
4. Persevering is a never give up attitude even though the problem is hard.
5. Collaboration is people working together to develop a good environment.

However, the common definition of computational thinking has some shortcomings and challenges. Peter J. Denning has already pointed out that the common definition is overly vague. He also claimed that computer scientists are worried about the confusion between computational thinking and computer science (Denning, 2017; Tedre & Denning, 2016). Computational thinking describes a small part of computer science. These shortcomings and challenges are valid and should be discussed in any computer science course. Nevertheless, computational thinking may be small but critical and must be learned.

2.2 Educational robotics

Educational robots familiarize children of all ages and help their knowledge of robots. In this section, the background of educational robotics is divided into 2 parts: fundamental of robotics and review of educational robotics.

2.2.1 Robotics

Robotics is a science and technology that is the integration of computer science and engineering. The aim of robotics is to design intelligent machines that can assist humans in their daily work, ensuring the safety of human work (Wikipedia, 2020d). The robots are divided into 6 types (Wikipedia, 2020c):

1. Mobile robots have the ability to move around in their environment and not be fixed at a position, e.g., an automatic guided vehicle (AGV).
2. Industrial robots used for manufacturing. Industrial robots are fully automated, programmable, and can move in three or more axes.
3. Service robots-semi-automated or autonomous to provide services beneficial to the well-being of humans and equipment, except for production.
4. Educational robots teach design, analysis, application, and operation of robots. Robots are made up of articulated robots, mobile robots, or autonomous vehicles.
5. Modular robots are a new generation of robots designed to increase robot utilization by creating a modular architecture.
6. Collaborative robots can safely and effectively interact with a human while working easy industrial tasks.

2.2.2 Research on educational robotics

The field of educational robotics includes many different facets, such as physical platforms, educational resources, and tangible technology. Educational robotics are commonly used in educational activities to transfer academic knowledge and skills related to Science, Technology, Engineering, and Mathematics (STEM) (Bargagna et al., 2019). Educational robotics provides a tangible way that students can easily send instructions to a robot then have their input validation without yet having to learn syntax (Shim, Kwon, & Lee, 2016).

The education illustrated the integration of educational robotics into the undergraduate computer science curriculum by using mobile robotics related to computer vision (Cielniak, Bellotto, & Duckett, 2012). There is a study that presented a finding which indicated that active, cooperative, and problem-based learning using mobile technology was suitable for both undergraduate and graduate robotics education (Riek, 2012). Another study shows the impact of educational robotics on children's technical, social, and science-related skills. The study depends on a two-point measurement (pre and post-test) and practical a multiple-choice questionnaire to evaluate the impact (Kandlhofer & Steinbauer, 2016).

Chang et al. believe educational robotics can help the kids in developing collaboration and communication, problem-solving abilities, critical thinking skills, and creativity among students (Chang, Lee, Chao, Wang, & Chen, 2010). Furthermore, teachers could completely include educational robotics in the young children's computer programming curriculum because the robots offer a better tinkering approach than computer monitors (Bers, Flannery, Kazakoff, & Sullivan, 2014). Thus, educational robotics are appropriate and have been practical to students of different age groups.

Figure 1 Lego Mindstroms

The state of the art for educational robotics during the first decade of the century usually focused on programming the robot from a personal computer. The most prominent example of such a system is the Lego Mindstorm, shown in Figure 1 (Lego, 2018). On the other hand, there are mobile applications such as Robot School (Great, 2015) and Swift Playgrounds (Apple, 2018), shown in Figure 2, where virtual robots are programmed and controlled instead of physical ones. Finally, with recent advances in technology, there are applications like Tickle (Tickle Labs, 2017), shown in Figure 3, which uses a block programming environment on an iPad to program physical robots, drones, and LEGO devices.

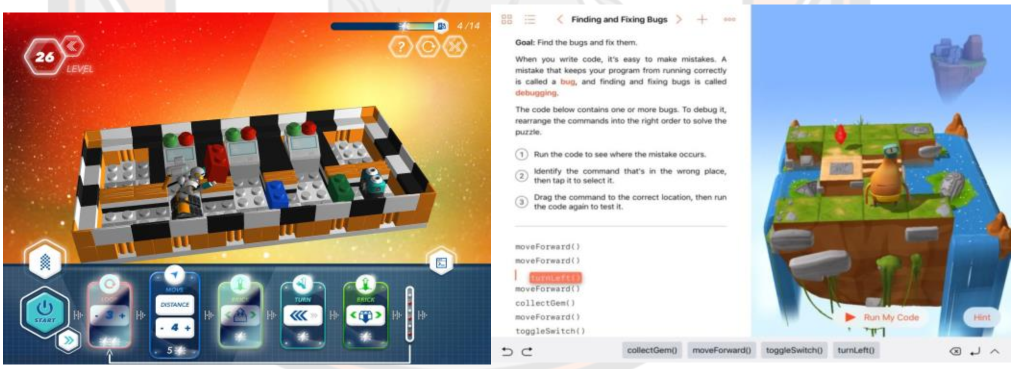


Figure 2 Robot school (left) and Swift playground (right).

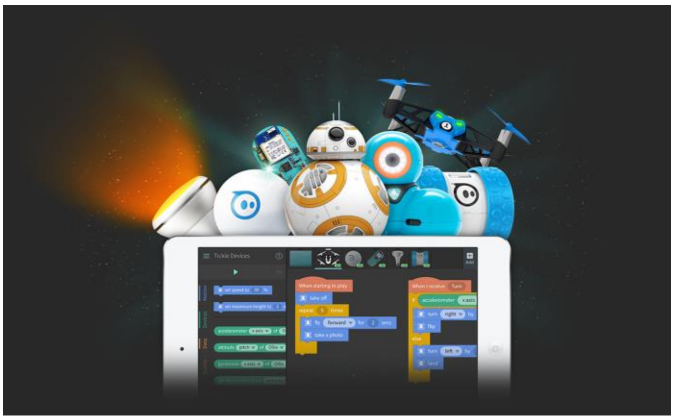


Figure 3 The Tickle app.

However, the arrival of educational robot use in schools has had a significant impact. For example, Nugent et al. used the robot together with geography technology to teach students about science, technology, engineering, and mathematics (STEM) (Nugent, Barker, Grandgenett, & Adamchuk, 2010). Another study by Alimisis et al. used educational robotics to identify new trends and challenges that focus on using robotics as a tool for creativity and other 21st century skills (Alimisis, 2013). Williams studied to estimate the impact of educational robotics on high school students' physics knowledge and scientific investigation skills (Williams, Ma, Prejean, Ford, & Lai, 2007). Chin et al. study developed an educational robot-based learning system that provides an attractive teaching application that combines multimedia objects with an educational robot. He examined the effect of the present learning system on student performance and motivation (Chin, Hong, & Chen, 2014). By no means educational robotics are the silver bullet. The tools are only as effective as the study plans and teaching materials.

2.3 Courseware

A courseware is generally used to describe educational materials. It could be a kit to teach, train, or tutorial the student most courseware is associated with technology-based materials. The term “courseware” is commonly referred to training for personal computers, software packages, or IT certification programs (Group, 2018; Rouse, 2018). Here are common courseware materials:

- instructor-led video or notes.
- self-directed computer-based training (CBT).
- interactive tutorials.
- live or webinar.

There are several available coursewares that promote computational thinking. The courseware in (Burbaitė, Drąsutė, & Štuikys, 2018) demonstrates how STEM-driven computer science (CS) education supports the development of computational thinking at the high school. In Thailand, a robotic training workshop was set up to promote computational thinking processes for the pre-engineering students (Hutamarn et al., 2017). The most famous courseware for computer science and computational thinking is code.org (Code.org, 2017; Kalelioğlu, 2015) where children learn how to write a code through a web application. The problems in code.org generally use famous children’s characters to narrate and teach each concept. This non-profit organization was first created to promote CS in K-12 students. While code.org offers a wide range of activities (both online and unplugged), the programming portion requires students to sit in front of their computers. This study wants to encourage the five computational thinking approaches: tinkering, creating, debugging, persevering, and collaboration. Playing with a tangible and mobile device in a group should encourage all five approaches especially collaboration. Gagne (instructionaldesign.org, 2018), an educational psychologist, in the 1960s, proposed nine important instructional design principles:

- gaining attention (reception).
- informing learners of the objective (expectancy).

- stimulating recall of prior learning (retrieval).
- presenting the stimulus (selective perception).
- providing learning guidance (semantic encoding).
- eliciting performance (responding).
- providing feedback (reinforcement).
- assessing performance (retrieval).
- enhancing retention and transfer (generalization).

These principles could be translated and applied to a modern-day learning environment. Today's courseware, according to (Penfold, 2016) should grab attention, present information, provide feedback, and enhance retention and transfer. The development of this study courseware will need to adhere to these principles.

Chapter 3
Methodology

The combination of tangible and mobile technologies created particularly for this study is called "Arducation Bot", based on the work of (T Phetsrikran et al., 2017; Titiphan Phetsrikran et al., 2018). Arducation Bot was designed to be an educational platform for improving primary school students' computational thinking ability. the previous version of the platform was found to have a many problems in both hardware and software as shown in Table 1. This section describes the technical aspects of Arducation Bot and, more generally, study implementation.

Table 1 Lists of the problem of the Arducation Bot from the previous version.

Hardware	Software
1. The robot did not walk in a straight. 2. The motor driver did not have enough power. 3. The batteries did not long life. 4. The Bluetooth was difficult to connect.	1. The commands were difficult to understand. 2. The application didn't know the status of the robot than connected 3. The application did not record answers and the time to solve each problem. 4. The UI was difficult to understand.

3.1 The specific history of the Arducation Bot

The roots of this current study began as an undergraduate study of the current author (T Phetsrikran et al., 2017), and the goal was to create a platform for teaching programming using a robot and an iPad rather than just a computer. In the beginning, the authors developed software to send commands to the robot, with communication between the software and the robot taking place via Bluetooth. Initial trials of the Arducation Bot prototype were conducted at St. Nicholas High School in Phitsanulok City, Thailand. The prototype was used to show students how mobile technology and educational robotics can be used to teach computational thinking in school. Initial trials of the Arducation Bot prototype showed that the use of mobile technology opens up alternative forms of interaction in the classroom that has the potential for highly collaborative activities. Figure 4 shows the first version of the Arducation Bot application and circuit diagram.

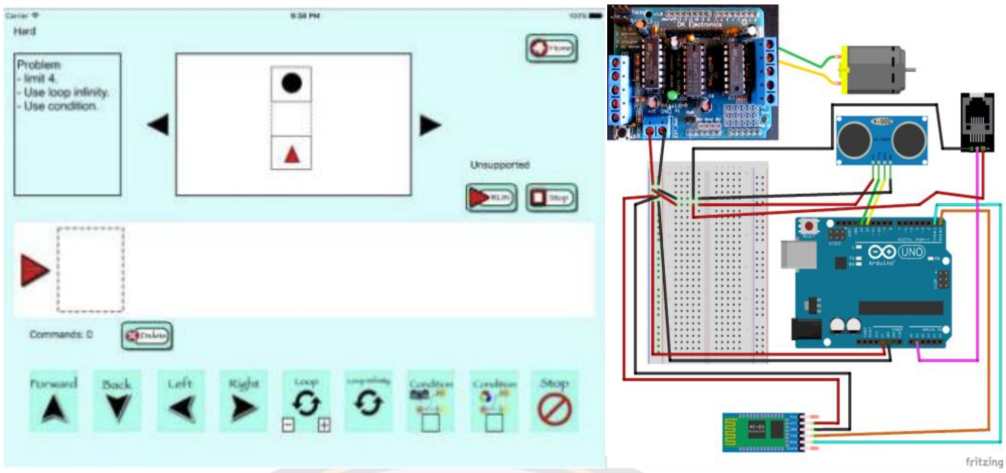


Figure 4 The initial version of the Arducation Bot application and circuit diagram.

After the first prototype was developed, areas for potential improvement and expansion were identified after reviewing students' comments on their experiences using the prototype. Additional ideas for improving on the first version came from a further reading of previous research in computational thinking and educational robotics.

The second version of Arducation Bot was thus developed as part of the current author's master's degree program. In the hardware, the first modification was to improve the robot's ability to move in a straight line. Other modifications to the components and structure of the robot included improving communication between the Bluetooth module and the iPad, enhancing the power of the motor driver, and lengthening the maximum battery life.

The software improvements for the second version of Arducation Bot included a redesign of the user interface to make it more intuitive and user friendly as well as the addition of functionality to record and report student input/responses and the time required to solve each puzzle. The completed second version of the Arducation Bot application and circuit diagram are shown in Figure 5. This second version was tested in 2018 with 180 high school students from Thailand and Japan at Chulabhorn High School in Phitsanulok City, Thailand. Results from that testing were then presented at the 2018 International Computer Science and Engineering Conference (ICSEC) held in Chiang Mai, Thailand (Titiphan Phetsrikran et al., 2018). The version two test results are shown in Figure 6.

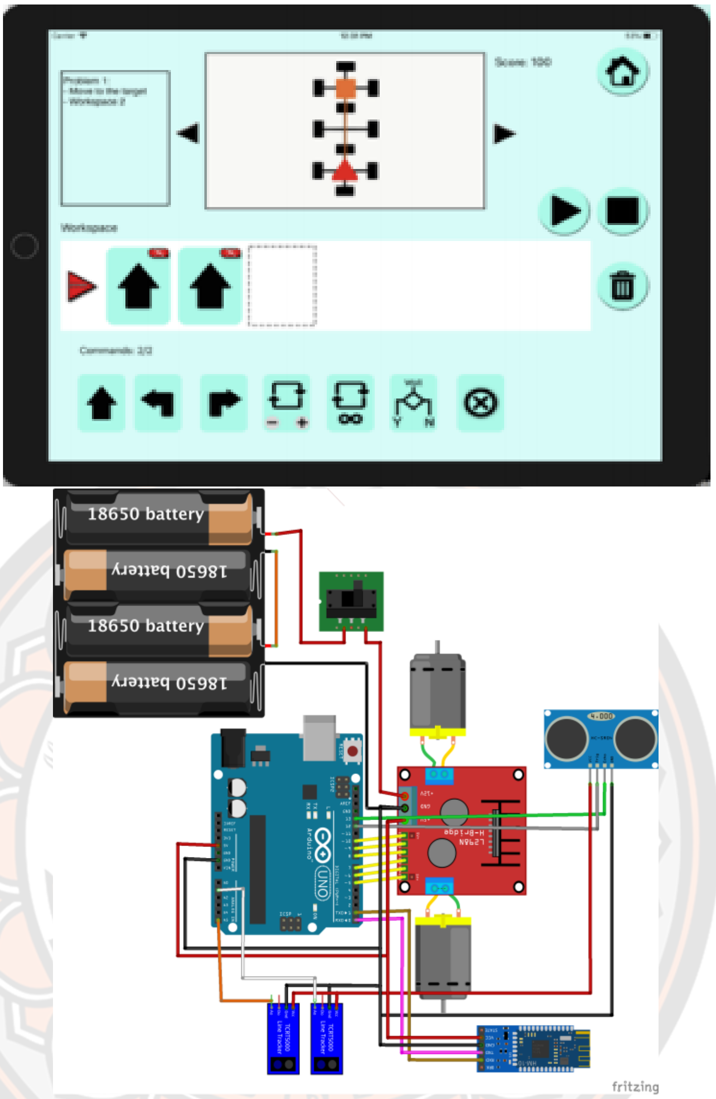


Figure 5 The second version of the Arducation Bot application and circuit diagram.

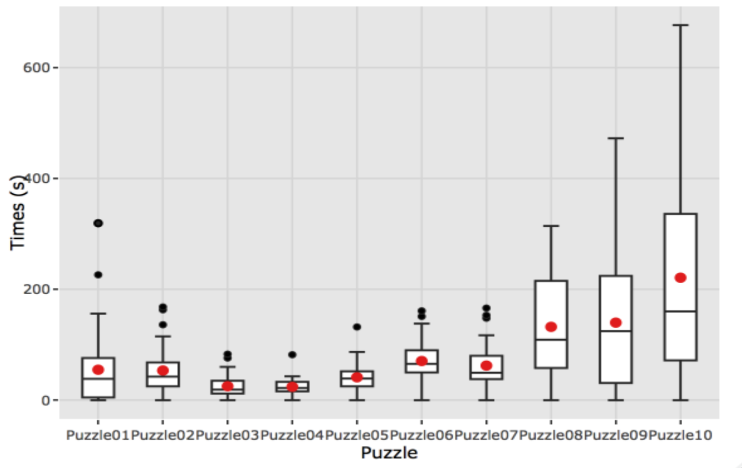


Figure 6 Results of Arducation Bot version two testing.

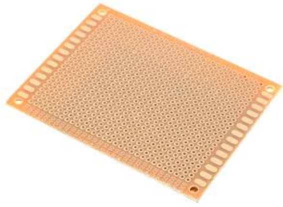
The newest version of Arducation Bot described in the current study is version three. Version three reflects further improvements in the design of the application and courseware. Specific details of these latest changes are explained in the following Hardware, Software, and Courseware section respectively.

3.2 Hardware

The components of the robot include an Arduino UNO R3, three IR sensors, an ultrasonic sensor, an L298n motor driver, two DC motors, an HM-10 Bluetooth BLE module, and a 3.7V li-ion battery. Everything is placed on a plastic frame, with the motor shafts serving as axles for the wheels. The circuit diagram is shown in Figure 7. The Arduino provides control and processing power for the robot. The two types of sensors, ultrasonic and IR, provide input feedback. The motors, controlled by the L298, provide physical actuation. Bluetooth provides communication between the robot and the iOS application the components of the robot are shown in Table 2.

Table 2 List of components for one Arducation Bot and their cost.

Name	**Image**	**Quantity**	**Price**
Arduino UNO R3		1	250 Baht
Motor driver L298n		1	115 Baht
Bluetooth HM-10 (BLE)		1	200 Baht
HY-SRF05 ultrasonic sensor		1	75 Baht
TCRT5000 infrared sensor		3	135 Baht

PCB board		1	12 Baht
Robot car chassis with 2 DC motor		1	250 Baht
Ultrafire 18650 3.7V 6800 mAh		3	150 Baht
18650 3 battery tough		1	25 Baht
			1,212 Baht

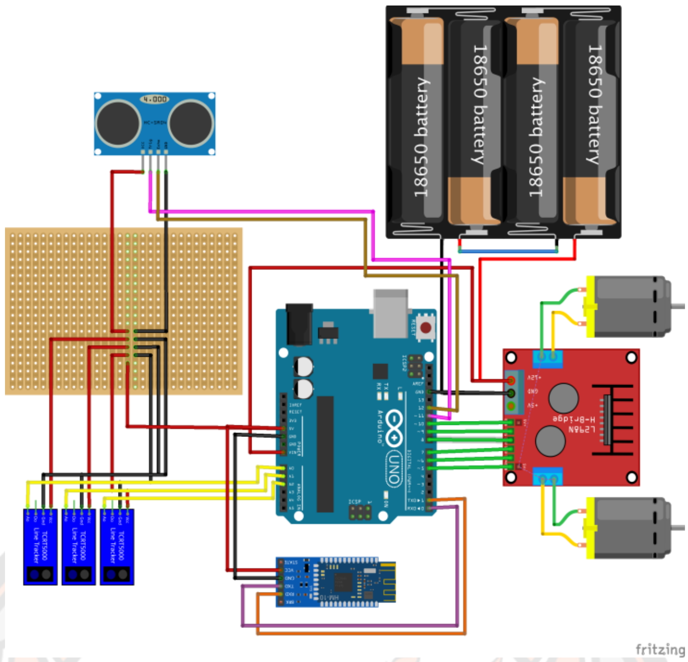


Figure 7 The Arducation Bot circuit diagram. Its components include an Arduino UNO R3, three IR sensors, an ultrasonic sensor, an L298n motor driver, two DC motors, and an HM-10 Bluetooth BLE module.

3.3 Rotary Encoder and PID Controller

During the course of improving the robot, rotary encoder and PID controller were studied as an option to assist the accuracy of the robot locomotion. This section explains the background of the two topics.

3.3.1 Rotary Encoder

The Rotary Encoder is a sensor that encodes the distance from its rotation and converts it into code in an electrical signal. These codes can be converted back to the desired values, i.e., if you want to measure distance, must be connected to a counter to display the distance. The display is the speed of RPM (Rotation Per Minute) based on the encoded signal to the electrical signal. It can divide a variety of encoding formats such as digital signals 0 and 1 or Binary Code and Gray Code (Wikipedia, 2020e).

3.3.2 PID Controller

PID (Proportional–Integral–Derivative) Controller normally used in a feedback control system. The values used in the calculation are the error values derived from the differences of the process variables and the desired values. The controller tries to minimize the error value by adjusting the process input signal (Wikipedia, 2020b). PID is dependent on three variables: Proportional, Integral, and Derivative. The PID Controller is a combination of the three variables according to (1).

$$MV(t) = P_{out} + I_{out} + D_{out} \quad (1)$$

where P_{out}, I_{out}, and D_{out} are the results of the output process from the PID controller. Each term is defined as described below.

a) Proportional (P)

The proportional will change to the proportion of the error value. The proportional response can be obtained by multiplying the error by the constant Kp, also known as the proportional gain, as explained in (2).

$$P_{out} = K_p e(t) \qquad (2)$$

where P_{out} is the output of the proportional.
K_p is the proportion gain which is adjustable.
e is the error value = r(t) - y(t).
t is the time.

The higher the proportional gain, the more change in error value. If it is too high, the system will become unstable. In contrast, the lower the proportional gain, the less response.

b) Integral (I)

Integrals are the proportions of the error size and the duration of the error. It is the sum of the errors in every moment. Integral of error gives the cumulative offset that should have been in the previous. The cumulative error is multiplied by the integral gain. The size of the integral term is determined by the integral gain (Ki) in (3).

$$I_{out} = K_i \int_0^t e(\tau)\, d\tau \qquad (3)$$

where I_{out} is the output of the integral.
K_i is the integral gain which is adjustable.
e is the error value = r(t) - y(t).
t is the time.
τ is the integral variable.

The integral accelerates the process to the desired point and removes residual errors by the use of proportional. However, integrals are in response to increased errors in the past and thus can overshoots.

c) Derivative (D)

The derivative is the rate of change of the process error that is calculated by the slope of the error every time and multiplied by the size of the Kd derivative. The magnitude of the derivative depends on the expansion rate of the Kd derivative in (4).

$$D_{out} = K_d \frac{de(t)}{dt} \qquad (4)$$

where D_{out} is the output of the derivative.
K_d is the derivative gain which is adjustable.
e is the error value = r(t) - y(t).
t is the time.

The derivative slows the rate of change of the control system's output, and with this effect it allows the control system to reach the desired point. Therefore, the derivative is used to decrease the size of the overload by the integral and improve the stability of the control system combination. However, the noise derivative in the control system is very sensitive to the infestation in the fault and can destabilize the process if the noise and differential gain is large enough.

The Proportions, Integrals, and Derivatives are combined to be the output u(t) of the PID controller. The final equation of the PID as shown in (5) and Figure 8.

$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt} \qquad (5)$$

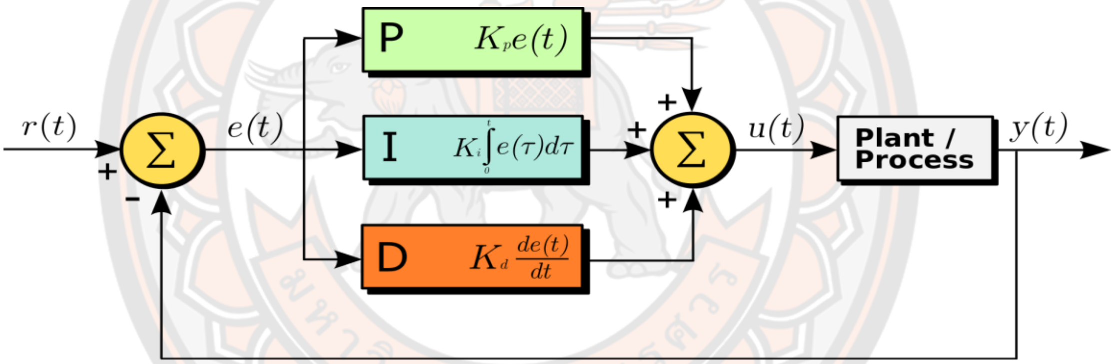


Figure 8 PID controller block diagram r(t) is a setpoint and y(t) is a process variable.

However, the PID controller theory was applied to improve the robot's locomotion stability. The encoder was installed on the motor. Motor speed encoder is used to measure the distance of one rotation as shown in Figure 9. This method can check the rotation and measure the distance.

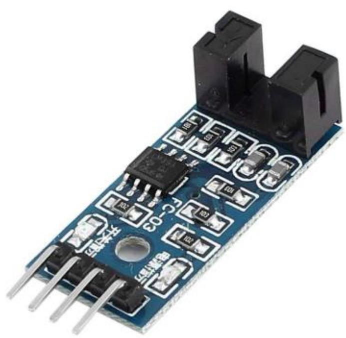


Figure 9 Lm393 Motor Speed Encoder

While using a normal DC motor with an encoder, it is difficult to control the rotation of both wheels in the same position. One solution is a motor with an encoder as shown in Figure 10. Using an encoder and a PID controller can solve the problem of spinning wheels in the same position and making the robot go straight. As the distances grow, the accumulative errors persist. Using a PID controller and a motor with an encoder can provide a better control over the direction of the robot. Maung et al. have used a DC geared motor with a PID controller to control the angular position of the motor and showed that the PID output is accurate to achieve the desired angle. This stability performance by using a PID controller can be applied to various control systems (Maung, Latt, & Nwe, 2018) and could present a major valuable improvement for this study.

Figure 10 Gear motor with encoder DC 12V.

The author tried to apply the encoder and PID controller. This method can decrease the robot's errors to a certain extent, but sometimes the robots walk off the track. Additional IR sensor was installed to improve the “follow the line” algorithm; however, Artyatha et al. (Artyatha, 2019) were able to achieve this functionality using the same principles.

3.4 Software

Two kinds of software have been developed for Arducation Bot: Arduino and Swift. The two main functions are to control the robot and to communicate with an iPad. The Swift-based iOS application provides an interface between the user and the robot. The user’s commands are sent from an iPad to the Arduino board through Bluetooth communication. The communication flow starts by the robot sending a “ready” message to the iPad indicating the robot’s readiness to receive a command. The iPad sends single commands to the robot, one at a time. After the robot finishes processing each individual command, it will send another “ready” message in order to receive the next command. Some commands are related to movement and others are related to obtaining information from the robot. For example, conditional commands ask the robot if there is an obstacle in front of the robot. The flowchart in Figure 11 also shows the algorithm between the Arduino and the iPad.

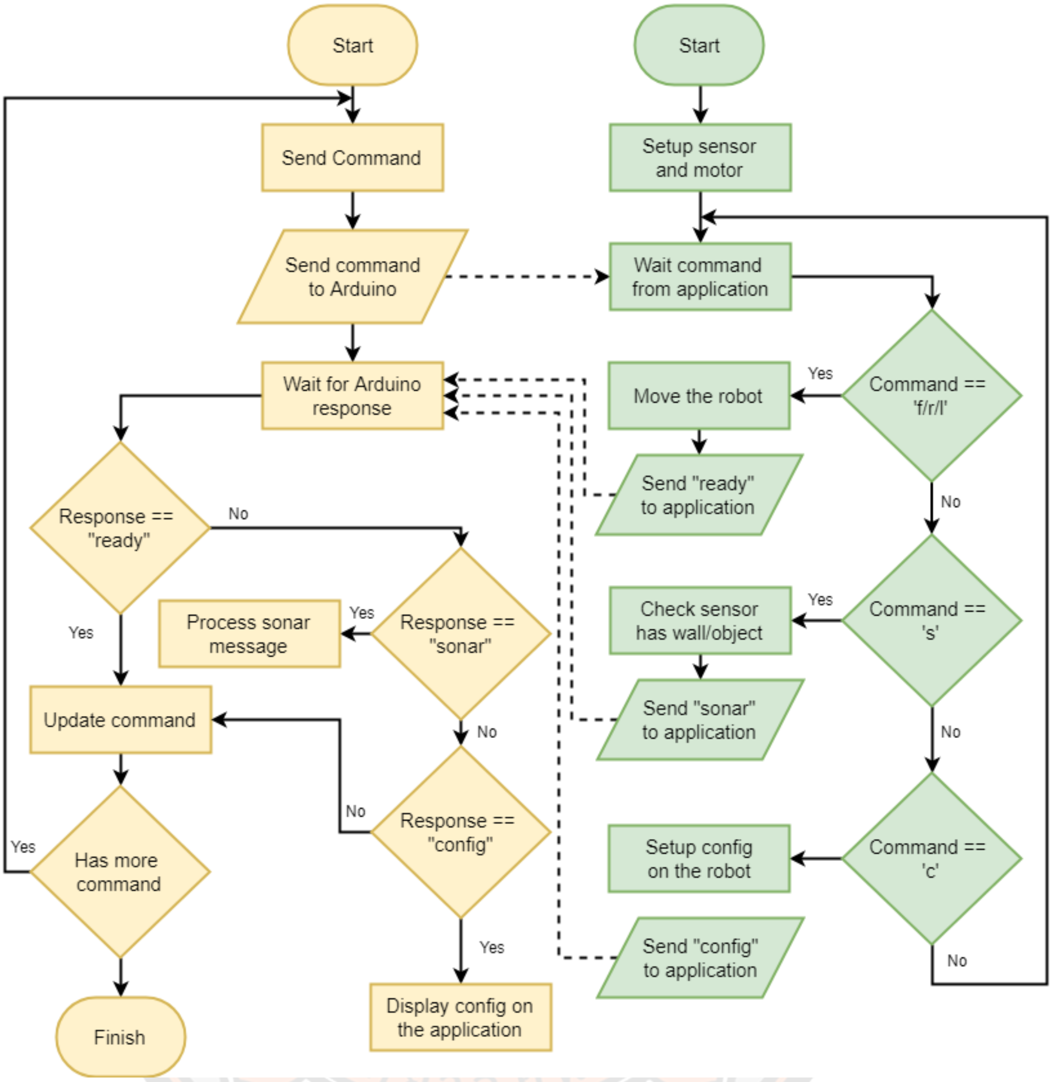


Figure 11 Flowchart of Arduino (right) and swift (left) programs.

The puzzles and the interface in the iOS application were modified from the curriculum and online sessions acquired through code.org. The target users of this platform are pupils above seven years of age. The UI/UX was designed to be intuitive and interactive, with users dragging and dropping the commands into the workspace. As shown in Figure 12, the application starts the children off with a simple puzzle that explains basic commands to control the robot. The application contains twenty different puzzles, divided into four units of increasing difficulty. The first unit requires the children to understand sequencing. The second unit teaches the concept of loops. The third unit is about conditionals. The fourth unit is a combination of a loop and a conditional. The design of the puzzles is explained in the courseware section below.

Figure 12 The iOS Application

3.5 Courseware

Arducation Bot is modeled on concepts and designs shared by many online courses. For example, two well-known online education websites are code.org and ScratchEd (https://scratched.gse.harvard.edu). Code.org (Code.org, 2017) is an online course in computer science. ScratchEd (Brennan, 2007) is an online education community where educators share their teaching experiences and methodologies, and it is not limited to computer science. Similar to the goals of these two websites, the Arducation Bot has been created to provide children with a course that teaches and improves computational thinking skills, skills which will help them learn computer science. The Arducation Bot platform consists of four units, each covering one topic with a series of target concepts, and the teaching of each target concept is built around one puzzle, as seen in Table 3.

Table 3 Structure of the Arducation Bot

			Concepts Taught in Each Puzzle					
Unit No.	Unit Topic	Puzzles	Logic	Algorithm	Decomposition	Patterns	Abstraction	Evaluation
1	Sequencing	Seq1	✓	✓			✓	✓
		Seq2	✓	✓			✓	✓

		Seq3	✓	✓			✓	✓
		Seq4	✓	✓			✓	✓
		Seq5	✓	✓			✓	✓
		Seq6	✓	✓			✓	✓
		Seq7	✓	✓			✓	✓
2	Loops	Loop1	✓	✓		✓	✓	✓
		Loop2	✓	✓		✓	✓	✓
		Loop3	✓	✓		✓	✓	✓
		Loop4	✓	✓		✓	✓	✓
		Loop5	✓	✓		✓	✓	✓
		Loop6	✓	✓		✓	✓	✓
		Loop7	✓	✓		✓	✓	✓
3	Conditions	If1	✓	✓	✓		✓	✓
		If2	✓	✓	✓		✓	✓
		If3	✓	✓	✓		✓	✓
4	Conditions with Loops	If-loop1	✓	✓	✓	✓	✓	✓
		If-loop2	✓	✓	✓	✓	✓	✓
		If-loop3	✓	✓	✓	✓	✓	✓

The four units are Sequencing, Loops, Conditions, and Conditions with Loops. In Unit 1 (Sequencing), children learn the concept of sequencing an activity into steps. In this unit, they also learn the basic commands and movements of the robot. In Unit 2 (Loops) children learn the concept of using a loop to repeat a procedure in order to accomplish a goal. In Unit 3 (Conditions) children learn the concept of expressing instructions based on a condition, using “If” and “Then”. For example, "IF you meet an obstacle THEN turn left, but if you do not meet an obstacle, then continue straight". This concept divides a big problem into a sequence of smaller tasks. In the above example, there are three such smaller tasks: 1) Check if there is an object. 2) If there is an object turn left. 3) If there is no object, continue forward. Finally, Unit 4 (Conditions with Loops) children learn to combine the two concepts of Conditions and Loops in order to reach a goal.

Children must learn various commands and think about the algorithm to solve these puzzles because there is a limited number of commands in each puzzle. For example, the first puzzle requires only two commands to solve. The loop concept is introduced with a puzzle that uses commands. Another concept that Arducation bot uses to design and improve children's skills is the conditional puzzle. This puzzle uses the conditions command when meeting the obstacles then stop. Three examples of puzzles shown in Appendix A illustrate the different themes. The design of all puzzles incrementally reveals the children's computational thinking concept as explained in the literature review section.

All three versions of the Arducation Bot were developed for the same purpose to improve your children's computational thinking skills. Developing from the first version to the latest version has always been a problem. Therefore, Arducation Bot latest version fixed and improved problems from versions one and two as shown in Table 4.

Table 4 Lists of the problem of the robot were improved from the first version to the latest version.

Problems	**First Version**	**Second Version**	**Latest Version**
Robot walk-straight	×	✓	✓
Bluetooth	×	✓	✓
Courseware	×	✓	✓
Design	×	×	✓
Components	×	×	✓

Chapter 4
Results and Discussion

4.1 Experiment

Figure 13 Testing of Arducation Bot at the Computational Thinking for Kids

The Arducation Bot was tested during four days in June 2019, as seen in Figure 13 with students from various primary schools (mostly near Phitsanulok City) during a one-day event called Computational Thinking for Kids, which was held four times-twice at Naresuan University and twice at St. Nicholas School in Phitsanulok City, as shown in Table 4. Each day, the students were split into ten groups, with four or five students in each group. Computational Thinking for Kids had two main sessions each day, called Unplugged (in the morning) and Arducation Bot (in the afternoon). At the beginning of the day, each student took a pre-test, and at the end of the day, as shown in Appendix B, they took a post-test. Data was collected from these two tests and processed to evaluate the difference in the computational thinking skills of each student before and after participating in the one-day event. The results are shown in Table 4.

Table 5 Testing events and statistical analysis of the pre-test and post-test

Event No.	**Date**	**Location**	**Participating Students (Age)**	**Student Count**	**Pre-test score**		**Post-test score**	
					$\bar{x}$	σ	$\bar{x}$	σ
1	3-June-19	Naresuan University	Grade 1 - 6 (6 - 12)	46	5.64	2.72	7.25	2.16
2	4-June-19	St. Nicholas	Grade 2 - 3 (7 - 8)	50	4.7	1.19	6.95	1.82
3	5-June-19	St. Nicholas	Grade 4 - 5 (9 - 10)	50	3.62	1.83	5.98	2.54

4	15-June-19	Naresuan University	Grade 1 - 6 (6 - 12)	31	5.48	2.82	7.6	1.69
				177	**4.86**	**2.14**	**6.95**	**2.05**

This data was derived from a total of 177 students participating over the course of the four days. It is known that students at Events 2 and 3 had never studied computational thinking before, but Events 1 and 4 may have included some students who had studied computational thinking previously. Looking at the mean points from the students' pre-tests and post-tests, the post-test scores were clearly improved, with a 43 percent increase over the pre-test scores. Standard deviations were 2.14 for the pre-test and 2.05 for the post-test.

The time required by each student to solve each puzzle was recorded in the iOS application. The puzzles are divided into four units: Sequencing, Loops, Conditions, and Conditions with Loops. By solving these puzzles, the student should obtain computational thinking skills. The skills are in logic, decomposition, algorithms, abstraction, patterns, and evaluation. Table 5 shows the Arducation Bot units, unit topics, and corresponding puzzles.

Each of the twenty puzzles requires the student to figure out one or more correct algorithms to move the robot from a starting point to a finishing point. When the student thinks they have figured out the correct algorithm(s) of the puzzle they are working on, they push the Run button. Then the robot will move according to their instructions (algorithms). However, if their algorithms are wrong, the student is informed and asked to try again.

In order to understand how the Arducation Bot platform improves a student's computational thinking skills, the time required by each student to correctly answer each puzzle was automatically recorded. Figure 14 summarizes the average time (of all 177 students) spent correctly answering each of the 20 puzzles

Table 6 The time required to solve each puzzle from 177 students.

Units	Topic	Puzzles. No.	Avg time	S.D.
1	Sequencing	1	15.99	11.74
		2	10.62	4.91
		3	6.45	3.53
		4	6.51	2.71
		5	5.21	2.17
		6	5.56	1.89

		7	5.71	2.01
2	Loops	8	22.76	13.63
		9	15.01	9.20
		10	22.96	12.22
		11	16.13	7.83
		12	20.65	10.17
		13	10.43	4.26
		14	7.35	3.99
3	Conditions	15	11.46	5.64
		16	9.00	4.53
		17	8.02	4.67
4	Conditions with Loops	18	13.34	7.93
		19	9.06	3.49
		20	6.62	2.83

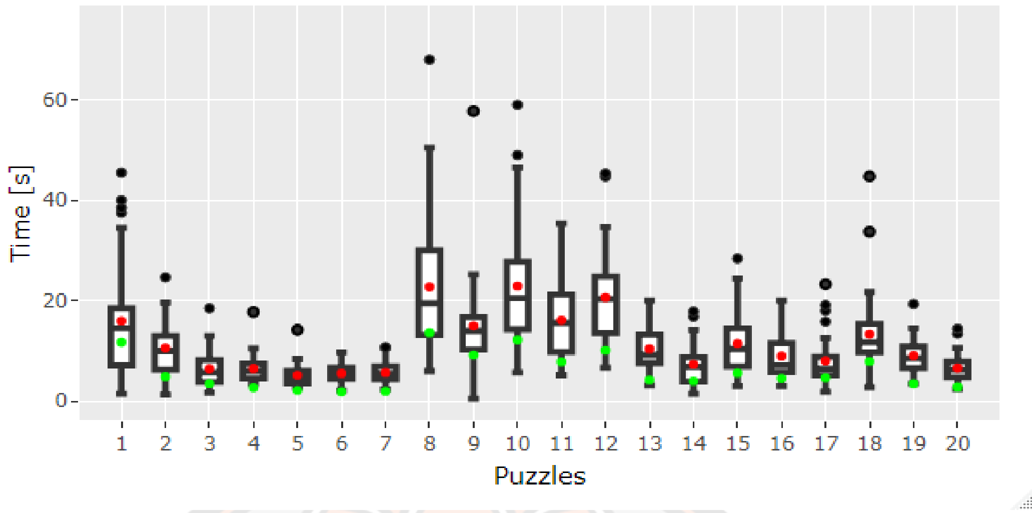


Figure 14 The average time required to successfully finish each puzzle

Interestingly, Units 1, 3, and 4 follow a similar pattern in Figure 14. In each of these three units, the student gradually used relatively more time per question at the start of the unit and relatively less time per question at the end of the unit. Only Unit 2 did not show this pattern. Since all the questions within one unit were approximately equally difficult, the shorter answering times as the unit progresses seems to show a clear improvement in the student's ability to understand the concepts in that unit. It is unclear just why Unit 2 did not follow the same pattern.

Sequencir	TRUE	1	3:24	1:03	right forward left forward right forward left forward forward stop
			3:51		
Sequencir	TRUE	1	4:30	0:39	right forward left forward right forward left forward forward stop
			9:02		
Loop: 1/7	FALSE	1	9:11		forward forward stop
Loop: 1/7	FALSE	2	9:38		loop(1) forward endLoop stop
Loop: 1/7	FALSE	3	9:56		forward loop(2) endLoop stop
Loop: 1/7	FALSE	4	10:24		loop(2) forward endLoop stop
Loop: 1/7	TRUE	5	11:09	0:22	loop(3) forward endLoop stop
			11:22		
Loop: 2/7	FALSE	1	12:08		loop(3) right forward endLoop stop
Loop: 2/7	TRUE	2	13:38	1:08	loop(3) forward endLoop right forward stop

			39:34:00		
Loop: 1/7	FALSE	1	40:06:00		forward forward stop
Loop: 1/7	TRUE	2	40:57:00	0:51	loop(3) forward endLoop stop
			44:04:00		
Loop: 2/7	FALSE	1	44:09:00		forward stop
Loop: 2/7	FALSE	2	44:40:00		loop(3) forward right endLoop forward stop
Loop: 2/7	FALSE	3	45:34:00		forward forward forward right forward stop
Loop: 2/7	TRUE	4	46:21:00	0:33	loop(3) forward endLoop right forward stop
			46:42:00		

Sequencir	TRUE	1	31:39:00	0:38	right forward left forward right forward left forward forward st
			35:58:00		
Loop: 1/7	FALSE	1	36:41:00		forward forward forward stop
Loop: 1/7	TRUE	2	38:00:00	1:19	loop(3) forward endLoop stop
			38:13:00		
Loop: 2/7	FALSE	1	39:09:00		loop(3) forward endLoop right stop
Loop: 2/7	FALSE	2	39:33:00		loop(0) forward endLoop right forward stop
Loop: 2/7	TRUE	3	39:49:00	0:32	loop(3) forward endLoop right forward stop

Figure 15 Conceptual misunderstanding within each unit.

With closer reexamination of the raw data (answers on each question), some students seemed to have a misconception at the beginning of each new study unit. Some students still used the concept of the previous unit without applying the new concept with the answer, as shown in Figure 15. In this particular example, Loop should have been applied. 9 out of 35 groups displayed such behavior, which accounted for 25.7% of all groups. By collecting and analyzing these data, this information can yield a deeper and better understanding of student's confusion.

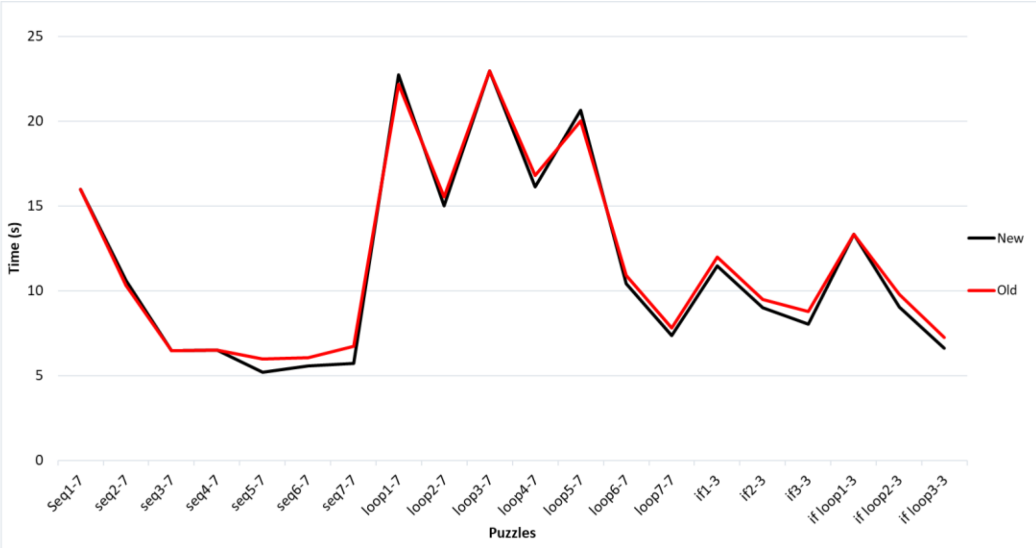


Figure 16 The average time per command of each puzzle.

The average time taken per problem with and without accounting for the inability to grasp the new concept can be seen in Figure 16. Both results are very similar. The root mean square errors (RMSE) were calculate to find the discrepancy between the two as shown in Table 7.

Table 7 The average time to calculation RMSE of twenty puzzles.

Puzzles	Actual (old)	Forecast (new)	Error	Square of error
seq1-7	15.9857	15.9857	0.0000	0.0000
seq2-7	10.6190	10.3238	-0.2952	0.0872
seq3-7	6.4500	6.4500	0.0000	0.0000
seq4-7	6.5071	6.5071	0.0000	0.0000
seq5-7	5.2057	5.9871	0.7814	0.6106
seq6-7	5.5551	6.0571	0.5020	0.2520
seq7-7	5.7143	6.7107	0.9964	0.9929
loop1-7	22.7571	22.1786	-0.5786	0.3347
loop2-7	15.0143	15.5357	0.5214	0.2719
loop3-7	22.9571	22.9571	0.0000	0.0000

loop4-7	16.1257	16.8014	0.6757	0.4566
loop5-7	20.6476	20.0310	-0.6167	0.3803
loop6-7	10.4286	10.8943	0.4657	0.2169
loop7-7	7.3524	7.8119	0.4595	0.2112
if1-3	11.4629	11.9857	0.5229	0.2734
if2-3	8.9952	9.4952	0.5000	0.2500
if3-3	8.0163	8.7602	0.7439	0.5534
if loop1-3	13.3357	13.3357	0.0000	0.0000
if loop2-3	9.0571	9.7905	0.7333	0.5378
if loop3-3	6.6204	7.2388	0.6184	0.3824
			RMSE	**0.5390**

The average time and the number of correct answers in each puzzle as shown in Figure 17. The average number of correct answers and the average time in each puzzle are related because in the first puzzle of each unit requires relatively more time while the last puzzle requires the least. This shows the clear improvement of student's understanding on computational thinking. With their developed skills, it enables them to use less time to solve problems.

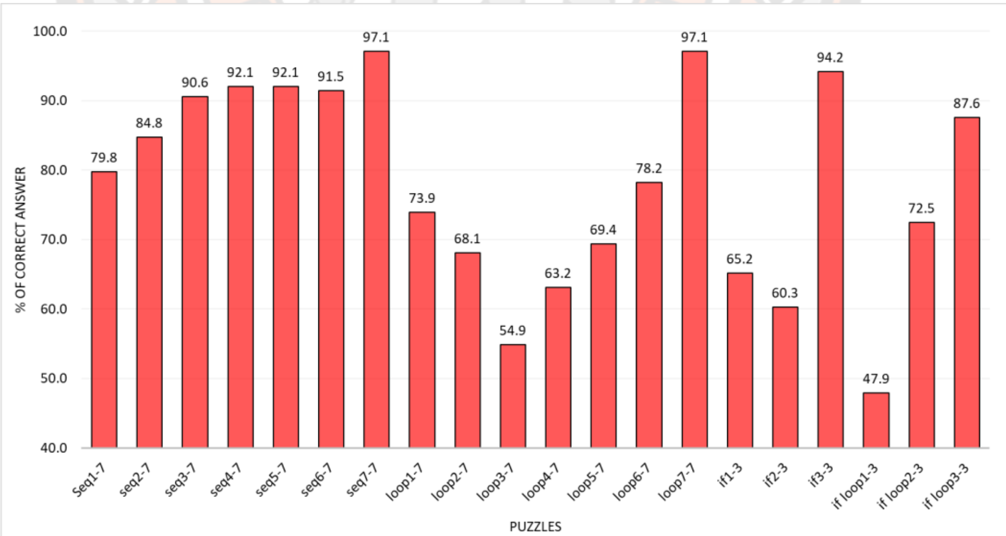
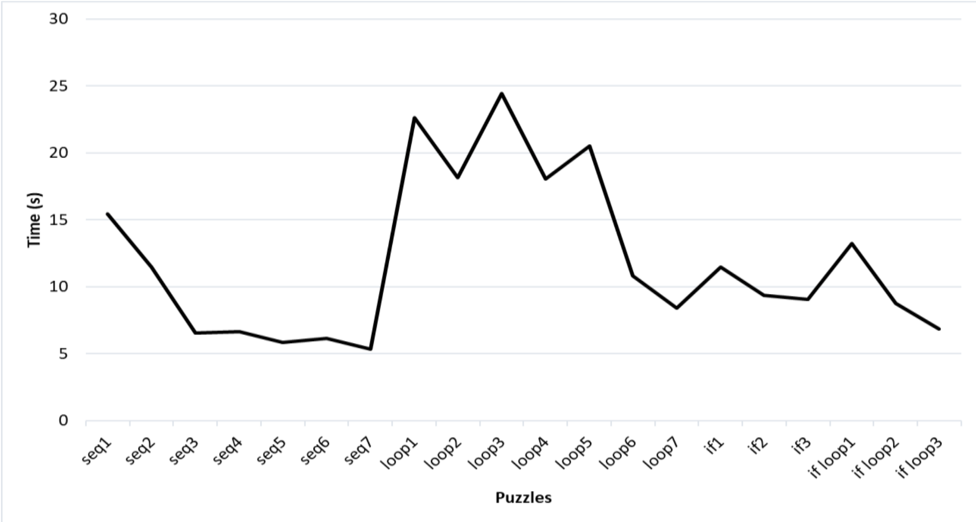
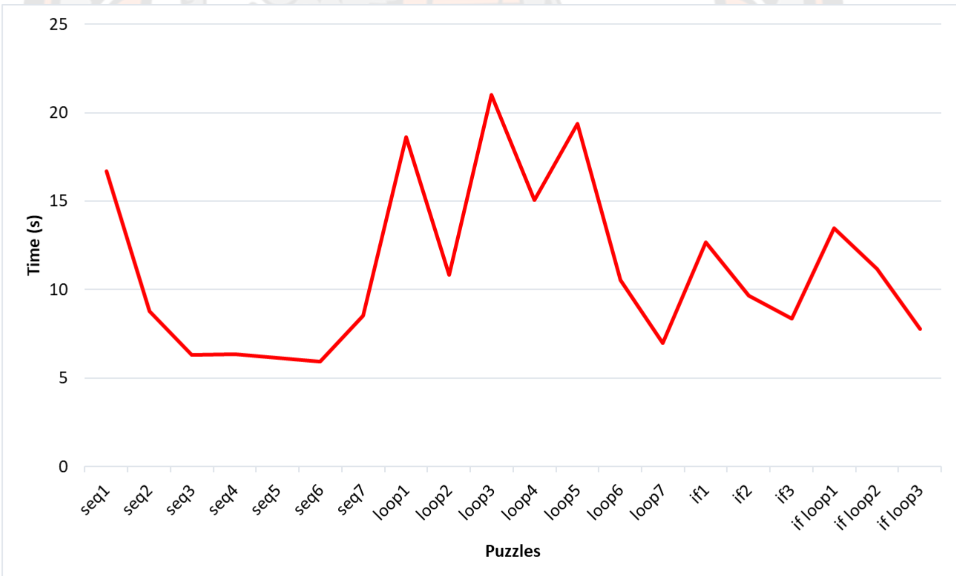


Figure 17 Percent of the correct answers to each puzzle.

It should also be pointed out that conditions varied on four different test days. During the two test sessions at St. Nicholas School, students were on their home turf so to speak and being supervised by their regular teachers. Therefore, the students tend to be relatively well behaved. On the other hand, during the two other test sessions at Naresuan University, students were visiting the campus only to take the test and were surrounded by a new environment full of novel stimuli, so they tended to be relatively less behaved and less focused on the task at hand. However, when the test results from the two locations are separated and compared, the results are strikingly similar to each other, as seen below in Figure 18. This similarity of outcome even in two different settings and atmospheres suggests that the test is valid and meaningful.



(a)



(b)

Figure 18 The average time required to successfully finish each puzzle at two different locations: a) at the student's own school and b) at Naresuan University

The differences between the student with prior experience in computational thinking against those without were investigated. When comparing the two groups of data, it was found that the group with students who had previously studied computational thinking requires slightly less time to achieve the correct answers with the exception of the condition unit as shown in Figure 19. Nonetheless, the courseware created for this study proved that students' computational thinking skills could be improved with or without prior knowledge. The Arducation Bot platform successfully improved the students' computational thinking skills, enabling them to better understand and solve various computational tasks.

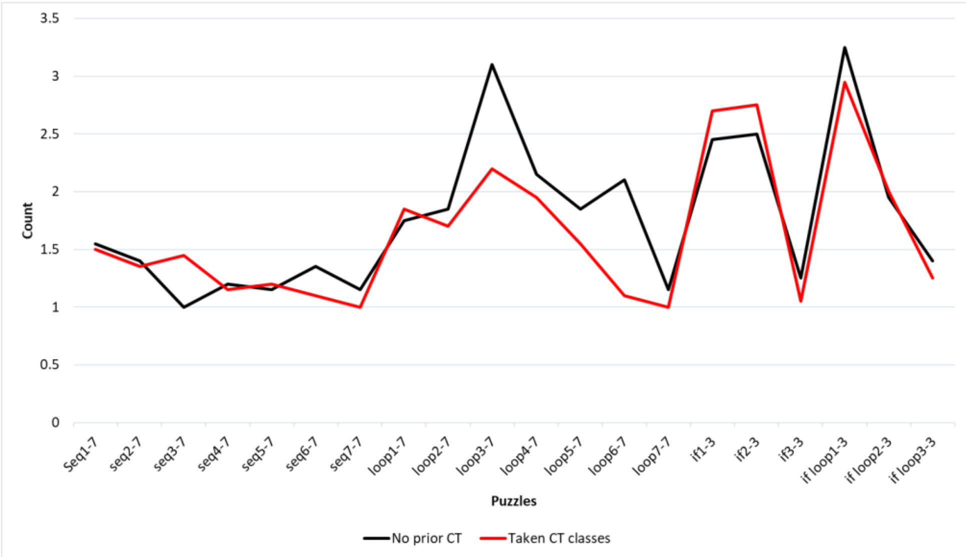


Figure 19 The number of average correct answers both groups: student no prior computational thinking and student has taken computational thinking classes.

4.2. Discussion

The results showed a total of 177 students taking the questionnaire. The mean points from students' pre-tests and post-tests. The post-test scores were improved by more than 40% increase over the pre-test. However, Standard deviations were 2.14 for the pre-test and 2.05 for the post-test are very high.

The data by each student to solve twenty puzzles was recorded in the application. Arducation Bot improved students' computational thinking skills. Looking at Units 1(puzzle1-7), 3(puzzle15-17), and 4(puzzle18-20) follow a similar pattern. In each of these three units, the student gradually used relatively more time per question at the start of the unit and relatively less time per question at the end of the unit. Only Unit 2 did not show this pattern. Since all the questions within one unit were approximately equally difficult, the shorter answering times as the unit progresses seems to show a clear improvement in the student's ability to understand the concepts in that unit. The problem in this study:

- The wires of the robots are tangled, causing it to put off and be electric shock.
- The Bluetooth of the robot must write UUID to select and connect with the iPad.
- The courseware can be increased and improved.

All these suggestions will be useful for the future development of this study.

Chapter 5
Conclusions

5.1 Conclusion

This study developed a tangible tool that utilizes mobile technology to create an educational platform in computational thinking. The results from 177 primary school students who participated in the Computational Thinking for Kids event have shown the potential of this courseware platform. A clear pattern of improved computational thinking was demonstrated by the pre-test and post-test scores and related data from the Arducation Bot. This platform presents a low-cost and intuitive teaching tool that can effectively develop skills in computational thinking and prepare students for computer science.

5.2. Limitations

The research on this study has been finished. The author has listed the limitations and mistakes of my thesis as follows:

- iPad of the application must iPad 2017 (gen 5) or later for AR support.
- Arducation Bot can connect to Bluetooth with many devices together but It needs to list the UUID of each robot.
- Loops commands are not able to use nested loops.

5.3. Future works

In this study, the author would like to suggest the following for the future development for the Arducation Bot.

5.3.1 Hardware:

- Use a motor with an encoder and PID controller, so that motors rotate and stop at the same time.
- Change the design of the robot to be more robust, kid proof, kid friendly, etc.
- Use Bluetooth modules that function for Android and iOS.
- Increase battery life for a full day.

5.3.2 Software:

- Add more puzzles for understanding the concepts of computational thinking.
- Improve the loops commands that can be nested loops.
- Add the function command to be more diverse to the puzzle design.

5.3.3 Augmented Reality

Another “future” work in progress for this study is AR. Augmented Reality (AR) is a technology that combines reality and virtual worlds created via software and devices, which is considered as creating another piece of information that is constitutive in the virtual world such as graphics, video, images 3D text and text to overlap with real-world images that present on the camera (Wikipedia, 2019). In this study, AR was used to increase user interest in robots. Near the top of the robot, there is an AR marker for the detection of the robot's location by the iPad to help with evaluation and debugging. In this way, the camera of the iPad can see a line model drawn by the movement of the AB robot, as shown in Figure 20.

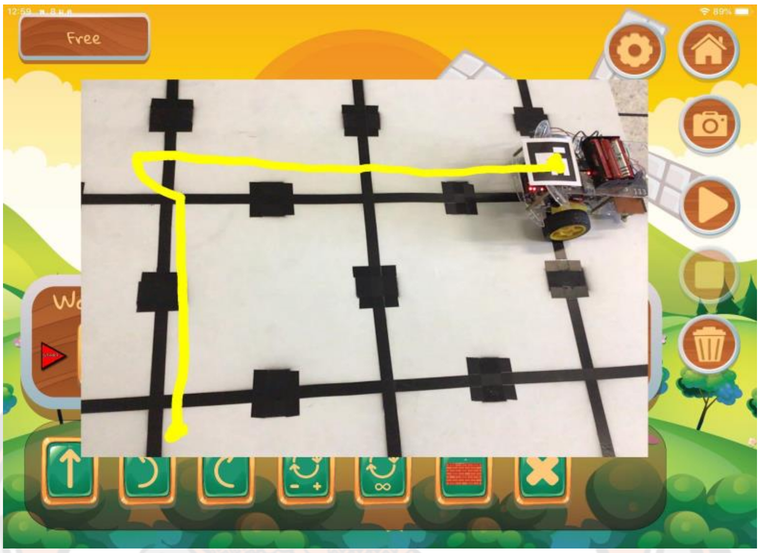


Figure 20 Simulation drawing line on application.

After the initial design and development AR for the Arducation Bot robot, this function was tested and evaluated. Two issues were identified. First, the iPad devices initially used for testing were older versions and they did not support AR. Switching to the newest iPad version resolved that issue. The second issue relates to the distance between the marker and the camera. If that distance is too far, then the camera is not able to detect the marker. Other AR studies do not experience this difficulty because, in those works, the distance was controlled or the marker did not move (Sittiyuno & Chaipah, 2019). This AR function for the Arducation Bot is not yet completed. More improvement should be done to help children improve debugging skills, which, in the author's opinion, is one of the most important skills in Computational Thinking.

REFERENCES

Alimisis, D. (2013). Educational robotics: Open questions and new challenges. *Themes in Science and Technology Education, 6*(1), 63-71.

Apple. (2018). Swift Playgrounds. Retrieved from https://www.apple.com/swift/playgrounds/

Artyatha, S. (2019). Arduino-based Educational Robotics with Android Mobile Application. วารสาร นวัตกรรม วิทยาศาสตร์ เพื่อการ พัฒนาอย่าง ยั่งยืน, *1*(1), 69-69.

Barefoot. (2014). Computational Thinking concepts and approaches. Retrieved from https://www.barefootcomputing.org/concept-approaches/computational-thinkingconcepts-andapproaches/.

Bargagna, S., Castro, E., Cecchi, F., Cioni, G., Dario, P., Dell'Omo, M., . . . Pecini, C. (2019). Educational robotics in down syndrome: a feasibility study. *Technology, knowledge and learning, 24*(2), 315-323.

BBC. (2017). "Introduction to computational thinking. Retrieved from https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72*, 145-157.

Brennan, K. (2007). ScratchEd. Retrieved from https://scratched.gse.harvard.edu/

Burbaitė, R., Drąsutė, V., & Štuikys, V. (2018). *Integration of computational thinking skills in STEM-driven computer science education*. Paper presented at the 2018 IEEE Global Engineering Education Conference (EDUCON).

Chang, C.-W., Lee, J.-H., Chao, P.-Y., Wang, C.-Y., & Chen, G.-D. (2010). Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school. *Journal of Educational Technology & Society, 13*(2), 13-24.

Chin, K.-Y., Hong, Z.-W., & Chen, Y.-L. (2014). Impact of using an educational robot-based learning system on students' motivation in elementary education. *IEEE Transactions on learning technologies, 7*(4), 333-345.

Cielniak, G., Bellotto, N., & Duckett, T. (2012). Integrating mobile robotics and vision with undergraduate computer science. *IEEE Transactions on Education, 56*(1), 48-53.

Code.org. (2017). Hour of Code: Moana. Retrieved from http://code.org/moana

Council., O. o. t. E. (2017). *The National Scheme of Education B.E. 2560-*

2579.

Dek-D. (2017). Computational Thinking. Retrieved from https://bit.ly/2EKGhCN

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM, 60*(6), 33-39.

Great, N. i. (2015). Robot School Prgramming. Retrieved from https://itunes.apple.com/th/app/id943154220

Group, I. (2018). What is Courseware. Retrieved from https://www.igi-global.com/dictionary/disruptive-technologies-and-education/6098

Hutamarn, S., Chookaew, S., Wongwatkit, C., Howimanporn, S., Tonggeod, T., & PANJAN, S. (2017). *A stem robotics workshop to promote computational* *thinking process of pre-engineering students in thailand: Stemrobot*. Paper presented at the The 25th International Conference on Computers in Education.

instructionaldesign.org. (2018). Conditions of Learning (Robert Gagne). Retrieved from https://www.instructionaldesign.org/theories/conditions-learning/

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior, 52*, 200-210.

Kandlhofer, M., & Steinbauer, G. (2016). Evaluating the impact of educational robotics on pupils' technical-and social-skills and science related attitudes. *Robotics and Autonomous Systems, 75*, 679-685.

Lego, T. (2018). LEGO Mindstorms. Retrieved from https://www.lego.com/en-us/mindstorms

Maung, M. M., Latt, M. M., & Nwe, C. M. (2018). DC motor angular position control using PID controller with friction compensation. *International journal of scientific and research publications, 8*(11), 149.

NECTEC. (2018). KidBright. Retrieved from https://www.nectec.or.th/news/news-pr-news/kidbright-coding.html/

Nugent, G., Barker, B., Grandgenett, N., & Adamchuk, V. I. (2010). Impact of robotics and geospatial technology interventions on youth STEM learning and attitudes. *Journal of Research on Technology in Education, 42*(4), 391-408.

Penfold, S. (2016). 9 instructional design principles for building an eLearning course. Retrieved from https://www.learnupon.com/blog/instructional-design-principles-building-elearning-course/

Phetsrikran, T., Harfield, A., Charoensiriwath, S., & Massagram, W. (2021). Arducation Bot: Computational Thinking Courseware with iOS Mobile Application and Educational Robotics. *ICIC Express Letters, Part B: Applications -- An International Journal of Research and Surveys (ICIC-ELB), 12*(1), 27-34. doi:10.24507/icicelb.12.01.27

Phetsrikran, T., Massagram, W., & Harfield, A. (2017). First steps in teaching computational thinking through mobile technology and robotics. *Asian International Journal of Social Sciences, 17*(3), 37-52.

Phetsrikran, T., Massagram, W., Phoka, T., & Harfield, A. (2018). *A Feasibility Study of Arducation Bot: An Educational Robotics and Mobile Application Kit for Computational Thinking Skills.* Paper presented at the 2018 22nd International Computer Science and Engineering Conference (ICSEC).

Reeve, E. M. (2013). Implementing science, technology, mathematics and engineering (STEM) education in Thailand and in ASEAN. *Bangkok: Institute for the Promotion of Teaching Science and Technology (IPST)*.

Riek, L. D. (2012). Embodied computation: An active-learning approach to mobile robotics education. *IEEE Transactions on Education, 56*(1), 67-72.

Rouse, M. (2018). courseware. Retrieved from https://whatis.techtarget.com/definition/courseware

Shim, J., Kwon, D., & Lee, W. (2016). The effects of a robot game environment on computer programming education for elementary school students. *IEEE Transactions on Education, 60*(2), 164-172.

Sittiyuno, S., & Chaipah, K. (2019). *ARCode: augmented reality application for learning elementary computer programming.* Paper presented at the 2019 16th

International Joint Conference on Computer Science and Software Engineering (JCSSE).

Tedre, M., & Denning, P. J. (2016). *The long quest for computational thinking*. Paper presented at the Proceedings of the 16th Koli Calling International Conference on Computing Education Research.

Tickle Labs, I. (2017). Tickle: Program robots, Drones, LEGO and Arduino. Retrieved from https://tickleapp.com/

Wikipedia. (2019). Augmented Reality. Retrieved from https://en.wikipedia.org/wiki/Augmented_reality

Wikipedia. (2020a). Educational robotics. Retrieved from https://en.wikipedia.org/wiki/Educational_robotics

Wikipedia. (2020b). PID controller. Retrieved from https://en.wikipedia.org/wiki/PID_controller

Wikipedia. (2020c). Robot. Retrieved from https://en.wikipedia.org/wiki/Robot

Wikipedia. (2020d). Robotics. Retrieved from https://en.wikipedia.org/wiki/Robotics

Wikipedia. (2020e). Rotary encoder. Retrieved from https://en.wikipedia.org/wiki/Rotary_encoder

Wikipedia. (2020f). Tangible user interface. Retrieved from https://en.wikipedia.org/wiki/Tangible_user_interface

Williams, D. C., Ma, Y., Prejean, L., Ford, M. J., & Lai, G. (2007). Acquisition of physics content knowledge and scientific inquiry skills in a robotics summer camp. *Journal of Research on Technology in Education, 40*(2), 201-216.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

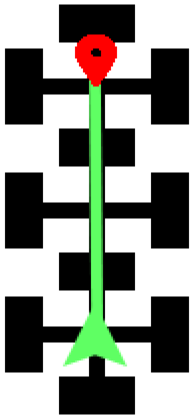
Wing, J. M. (2014). Computational thinking benefits society. Retrieved from http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

Appendix

Appendix A: Puzzles

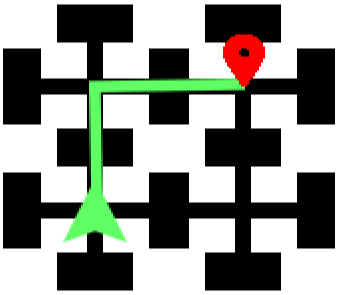
Unit 1: Sequencing: This unit introduces the concept of sequencing.

Puzzle 1



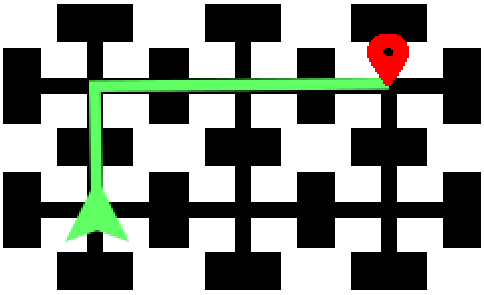
Puzzle 1 teaches the basic locomotion in which the answer to this one is go forward twice.

Puzzle 2



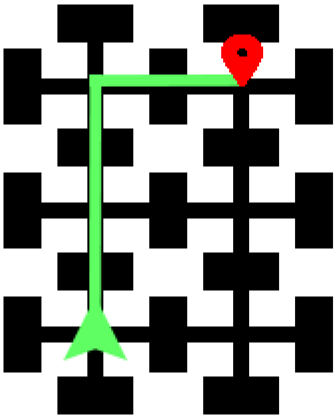
Puzzle 2 teaches how to combine more than one commands and place them in order. This answer is Forward, Right, and Forward.

Puzzle 3



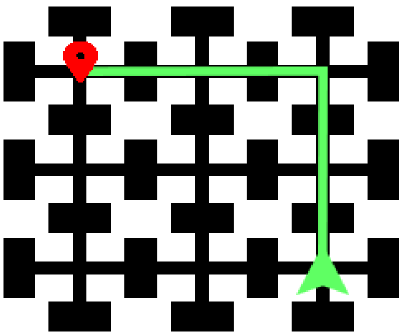
Puzzle 3, similar to Puzzle 2, is an understanding of the programming order. This answer is Forward, Forward, Right, and Forward.

Puzzle 4



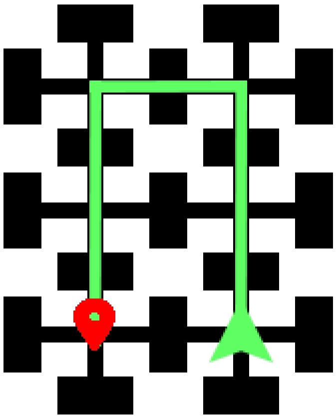
Puzzle 4 switches commands between Forward and Right to check the user's understanding of the sequence. This answer is Forward, Forward, Right, and Forward.

Puzzle 5



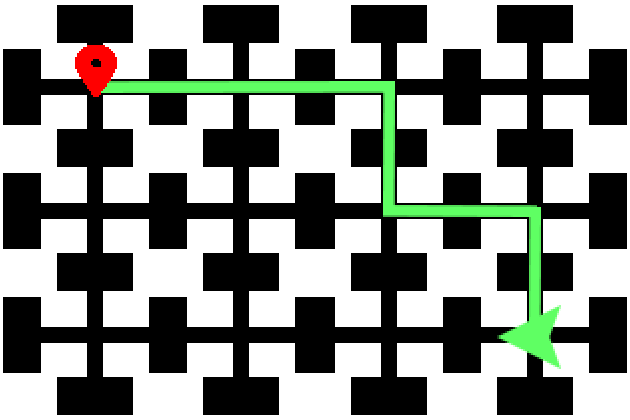
Puzzle 5 changes the robot's direction of rotation to improve the understanding of basic commands. This answer is Forward, Forward, Left, Forward, and Forward.

Puzzle 6



Puzzle 6 is similar to Puzzle 5 but with more turns. This answer is Forward, Forward, Left, Forward, Left, Forward, and Forward.

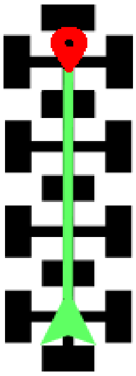
Puzzle 7



Puzzle 7 increases the complexity and uses all the basic commands to examine the understanding of the users. This answer is Right, Forward, Left, Forward, Right, Forward, Left, Forward, and Forward.

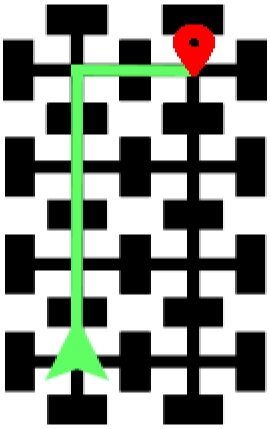
Unit 2: Loops: This unit introduces the concept of using a loop to repeat a procedure in order to succeed a goal.

Puzzle 8



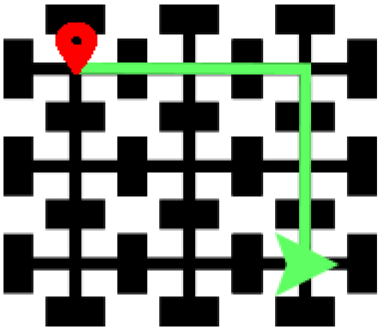
Puzzle 8 teaches users to use the loop command that repeats the function limited by the number of commands. This answer is Loop(3)(Forward). Number of commands 2.

Puzzle 9



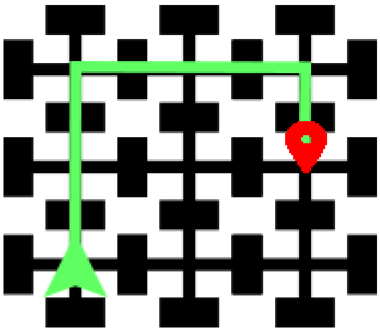
Puzzle 9 adds more command to check the comprehension of placing commands between in and out of the loop. This answer is Loop(3)(Forward), Left, and Forward. Number of commands 4.

Puzzle 10



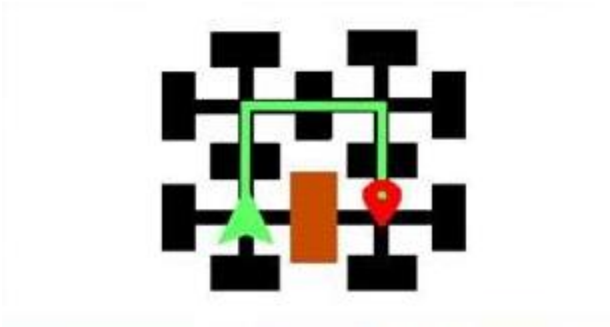
Puzzle 10 is similar to Puzzle 5, but changes the position of the robot and uses limited commands to increase the understanding that a similar pattern can be found. This answer is Loop(2)(Left, Forward, and Forward). Number of commands 4.

Puzzle 11



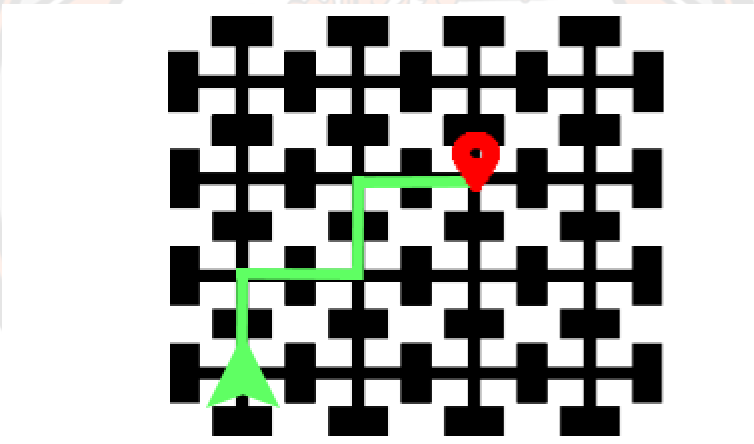
Puzzle 11 changes the direction of the robot from Puzzle 10 and adds one command to check for confusion between patterns and one more command. This answer is Loop(2)(Forward, Forward, Right), and Forward. Number of commands 5.

Puzzle 12



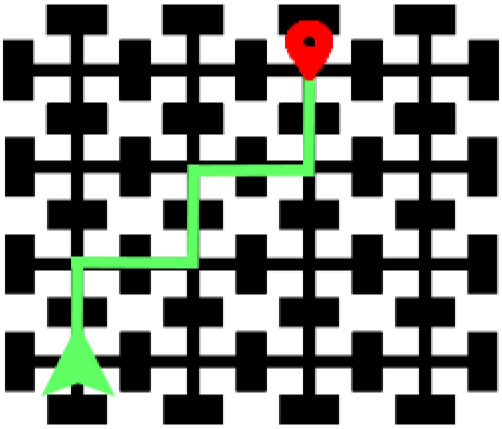
Puzzle 12 increases users' understanding in pattern finding, but more specific. At the finish point, the robot needs to rotate one more time. This answer is Loop(3)(Forward, and Right). Number of commands 3.

Puzzle 13



Inspired by Puzzle 7, Puzzle 13 requires a similar movement to Puzzle 7, but uses fewer commands to check users' understanding of pattern division. This answer is Loop(2)(Forward, Right, Forward, and Left). Number of commands 5.

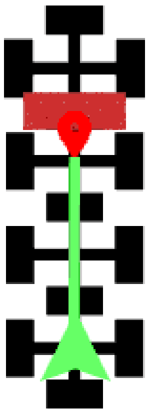
Puzzle 14



Puzzle 14, similar to Puzzle 13, can be divided into patterns of on loop and one more command outside of a loop. This answer is Loop(2)(Forward, Right, Forward, Left), and Forward. Number of commands 6.

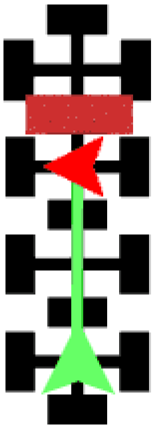
Unit 3: Conditions: This unit introduces the concept of conditional using "if".

Puzzle 15



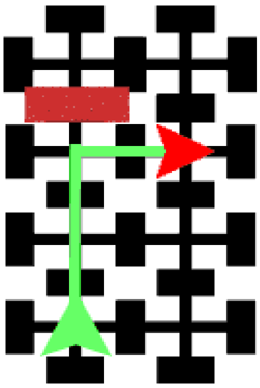
Puzzle 15 teaches the if command. It requires the user to decompose the concept, which has 2 sub-tasks: (1) if robot finds an obstacle then stops and (2) if robot does not find obstacle, continue moving forward. This answer is Forward, Forward, Condition(true)(stop), and Forward.

Puzzle 16



Puzzle 16 is similar to Puzzle 15, it requires increasing user understanding of if command and decomposition concept. This answer is Forward, Forward, Condition(true)(Left), and Right.

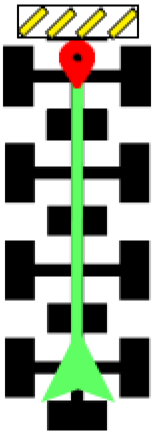
Puzzle 17



Puzzle 17 adds commands to increase a bit difficulty and check comprehension of placing between in and out of if commands. This answer is Forward, Forward, Condition(true)(Right, Forward), and Forward.

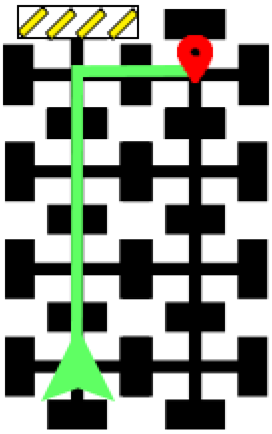
Unit 4: Conditions with Loop: This unit integrates the knowledge from the three previous units: Sequencing, Conditions, and Loops in order to reach our goal. This lesson adds a special command -- infinity loops because it requires the user to learn to solve problems if they do not know how many commands the puzzle is required.

Puzzle 18



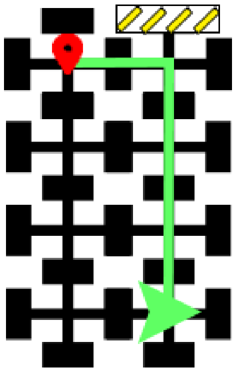
Puzzle 18 requires the combine between infinity loop and if commands without the user knowing where the obstacles are. This answer is Loop(infinity)(Condition(true)(stop), and Forward)

Puzzle 19



Puzzle 19 needs to increase the usability between infinity loop and if command and add a stop command. Since this puzzle uses an infinity loop, if the stop command is not used, the robot will not stop. This answer is Loop (Infinity)(Condition(true)(Right, Forward, stop), and Forward).

Puzzle 20



Puzzle 20, similar to Puzzle 19, increases the understanding of how to use commands and in the beginning. If the robot position changes, the user can still solve the problem using loops. This answer is Left, Loop(infinity)(Condition(true)(Left, Forward, stop), and Forward).

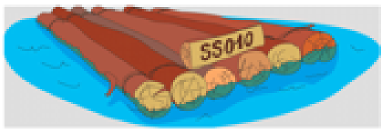
Appendix B: Pre-test and Post-test

Pre-Post Test

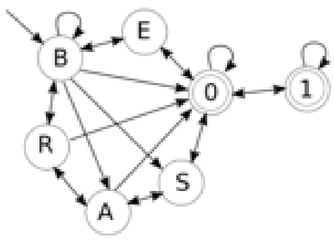
Name(s) ______________________________ Date ____________________

Part 1

Question 1



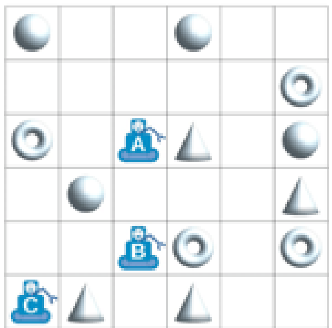
เจ้าหน้าที่กรมป่าไม้ ต้องการให้ผู้ประกอบการแพไม้ไผ่ขึ้นทะเบียนแพ เพื่อความสะดวกและปลอดภัยกับนักท่องเที่ยว นั่นหมายความว่า แต่ละแพจะต้องทำการจดทะเบียนแพที่มีตัวอักขระไม่ซ้ำกัน อักขระเหล่านี้ประกอบไปด้วยตัวอักษรและตัวเลขดังแผนภูมิด้านล่าง โดยทะเบียนแพจะต้องขึ้นต้นด้วยตัวอักษร B และสิ้นสุดด้วยตัวเลข 0 หรือ 1



คำถาม: ทะเบียนแพในข้อใดต่อไปนี้ ไม่ถูกต้องตามหลักการที่กำหนด

☐ BB0001 ☐ BBB100 ☐ BBB011
☐ BB0100 ☐ BR00A0 ☐ BSA001
☐ BE0S01

Question 2



ในโกดังเก็บของแห่งหนึ่ง หุ่นยนต์สามตัวทำงานร่วมกันเป็นทีม

เมื่อทีมได้รับคำสั่งทิศทาง (N, S, E, W) หุ่นยนต์ทั้งหมดในตารางจะเคลื่อนที่หนึ่งช่องในทิศนั้นๆ พร้อมกัน

หลังจากที่ปฏิบัติตามคำสั่งชุดคำสั่ง หุ่นยนต์จะทำการเก็บของที่พบในช่องสุดท้ายของคำสั่ง

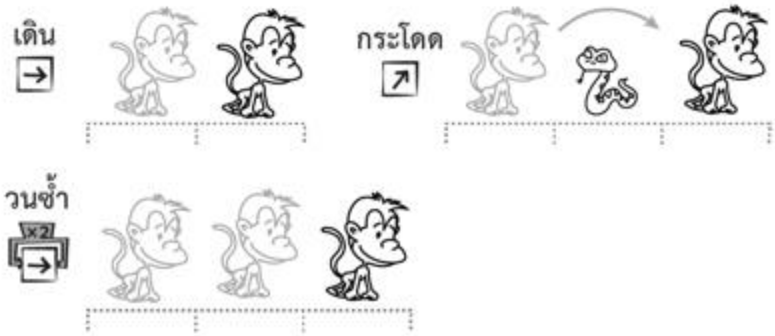
ยกตัวอย่างเช่น ถ้าเราให้ชุดคำสั่ง N, N, S, S, E กับหุ่นยนต์ทีมนี้ หุ่นยนต์ A จะหยิบกรวย หุ่นยนต์ B จะหยิบแหวน และหุ่นยนต์ C จะหยิบกรวย

คำถาม: ชุดคำสั่งในข้อใดต่อไปนี้ที่จะทำให้ทีมหุ่นยนต์หยิบลูกบอล กรวย และแหวน อย่างละหนึ่งชิ้น

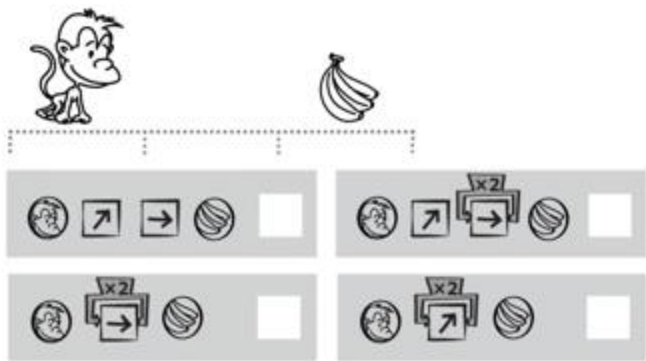
☐ N, E, E, E ☐ N, E, N, E ☐ N, E, E, S, E ☐ N, N, S, E, N ☐ N, E, E, S, W

Part 2

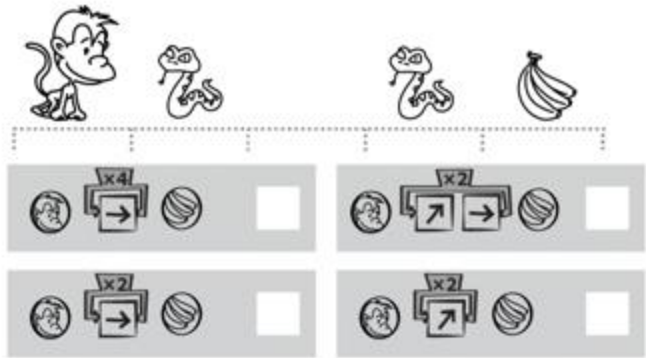
Commands (บล็อกคำสั่ง)



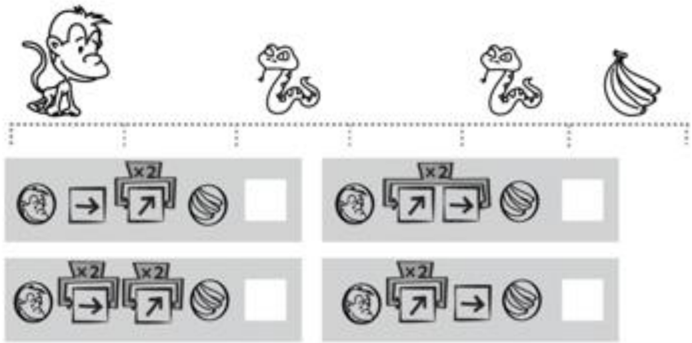
Question 3



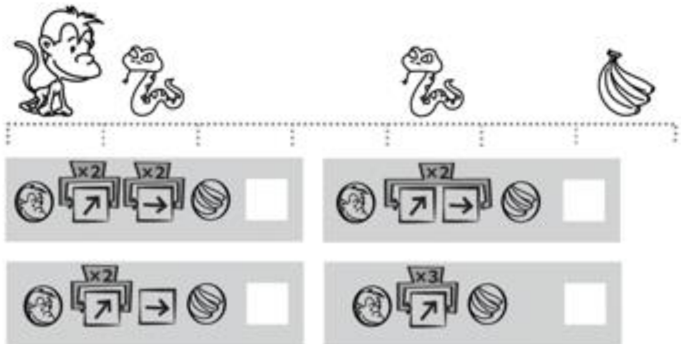
Question 4



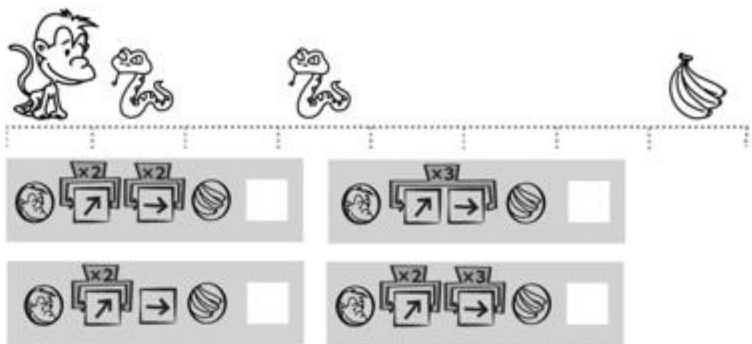
Question 5



Question 6



Question 7



Part 3

Question 8

ธงถัดไป คือ

Answer _______

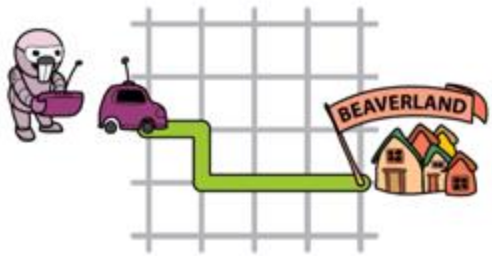
Question 9

ตัวเลขถัดไป คือ

1, 4, 13, 40, 121, ?

Answer _______

Question 10



โปรแกรมไหน ที่จะบังคับรถไปยัง BEAVERLAND?

A.	B.	C.	D.
Forward 1	Forward 3	Forward 3	Forward 1
Left	Right	Left	Right
Forward 1	Forward 1	Forward 1	Forward 1
Right	Left	Right	Left
Forward 3	Forward 1	Forward 1	Forward 3

Answer _______

A Feasibility Study of Arducation Bot

An Educational Robotics and Mobile Application Kit for Computational Thinking Skills

Titiphan Phetsrikran, Thanathorn Phoka, Antony Harfield, and Wansuree Massagram
Department of Computer Science and IT, Faculty of Science, Naresuan University, Phitsanulok, Thailand 65000
Email: wansureem@nu.ac.th

***Abstract*—The purpose of this study is to develop and test a pedagogical platform for teaching computational thinking through tangible and mobile technologies. Arducation Bot combines educational robotics and mobile application to create an intuitive and less intimidating approach in computational thinking and computer science. A large-scale testing of Arducation Bot platform was conducted with 180 high school students from Thailand and Japan. A strong pattern of improved computational thinking was shown in the collected data. Arducation Bot shows its universal appeal or suitability for pupils with different nationalities and backgrounds. Teachers and educators could potentially benefit from applying this mobile application and tangible technology platform to help students understand computational thinking skill — an imperative skill for the 21st century.**

***Index Terms*—computational thinking, educational robotics, mobile technology, tangible technology, pedagogy**

I. Introduction

Computational thinking is a thought process that leads to the development of skill sets and techniques to solve problems. There is a misconception when refers to computational thinking. Generally, computational thinking is understood as "thinking like a computer". However, computational thinking is the skill to solve any problem methodically, systematically, and logically. Computational thinking skill helps breaking down the problem into smaller and more manageable parts, then find any patterns and use abstraction to identify possible solutions. Traditionally, computer scientists and software developers use computational thinking for the basis of their codes or computer programs [1]. Not only computational thinking help solving the problems in computer science and mathematics, but also in everyday life [2]. Therefore, there has been a big push in teaching computational thinking in school at a very young age from several Thai agencies [3–5]. The current "Thailand 4.0" policy of the Royal Thai government involves nurturing a "Learning Society", which includes a transformation from standardized and duty-driven to personalized and passion-driven learning, and fact-based and passive learning to idea-based and active learning. The government has realized the country's reliance on technology and discrepancy of citizen's readiness. With this regard, Thailand has finally introduced Computational Science as a compulsory in the country's elementary and high school curriculum.

This proposed study could contribute to the transformation by promoting new styles of learning consistent with the aims of a learning society, i.e. active learning and passion-driven. Furthermore, computers are ubiquitous and contribute to almost every part of our lives. Encouraging young people to be interested in computing and building their skills from an early age is a challenge if done in an uninteresting medium through the traditional method of teaching. Thus by applying tangible technology, children can interact with objects and make learning more approachable and exciting [6].

The purpose of this study is to combine tangible and mobile technologies to develop an easy to understand pedagogical tool for computational thinking improvement. The tool should be universally suitable for pupils with different ethnics and backgrounds. The tangible technology in this study has been proposed in a form of an Arduino-based robotic car that the students can touch and tinker with. The mobile technology in this study is an application on the iOS platform which includes puzzles to be algorithmically solved. The difficulty of the puzzles increases as the child progresses. We have reported a successful first step along with its pitfalls in [7]. This study attempted to improve the Arducation Bots platform and courseware. The following sections describe the background, the technical aspects, the experimental results of the study. The discussion about the practicality of using educational robotics and mobile application is provided towards the end of the paper.

II. Background

The inspiration of this study comes from the concepts of computational thinking and educational robotics. This section explains theories and how they have traditionally been applied in pedagogy.

A. Computational Thinking

Computational thinking is a basic skill for everyone, not just programmers or computer scientists. Without knowingly, our daily life constantly requires logical, analytical and problem-solving skills. Generally, computational thinking include but not limited to these following concepts: logic, algorithm, decomposition, pattern, abstraction, and evaluation [8–12]. These thought processes and approaches are useful not only in software engineering but in many other domains as well. A complex problem might seem difficult to solve at first. But computational thinking takes that complex problem and breaks it down into a series of small, more manageable problems. This step is called "decomposition". Each of these smaller problems can then be considered individually and solved methodologically with logic. The noticing of similarity between problems which have previously been solved previously is called "pattern recognition". The focusing of the important details while ignoring irrelevant information is called "abstraction". When simple steps or rules to solve the smaller problems are designed, that is called "algorithms". Finally, making judgment whether the solution is valid is "evaluation". These computational thinking concepts can be taught through these five approaches: tinkering, creating, debugging, persevering, and collaborating [13].

However, the broad definition of these cognitive skills and pedagogical methods to computational thinking have faced many criticism and challenges. Some has criticized it as too vague. Some computer scientists worry about the confusion between computational thinking and computer science education. Computational thinking represents a small part of the computer science field. Others worry that the social, ethical, and environmental implications of computer technology do not get emphasized enough if educators only focus on computational thinking. These criticism and challenges are valid and should be addressed in any computer science curriculum.

B. Educational Robotics

Robotics kits, programming software, and computer have been employed as teaching aids. This type of hands-on learning is called educational robotics which creates a non-traditional learning environment for students. There have been many reviews on educational robotics and its effectiveness [14–16]. Chang et al. [17] believe in the deployment of educational robotics to help the students in developing collaboration and communication, problem-solving abilities, critical thinking skills, and creativity among students. Teachers could effectively include educational robotics in the young children computer programming curriculum because the robots offer better tinkering approach than computer monitors [18]. At a college level, educational robotics have also been used in older students as presented in [19], which set up a robotics training workshop to promote computational thinking process for pre-engineering student. Educational robotics are suitable and have been applied to students of different backgrounds and age groups.

III. Design and Implementation

The combination of tangible and mobile technologies created specifically for this project is called "Arducation Bot". It was designed to be a teaching platform for introducing computational thinking concepts to children [7]. Thhis previous platform faced several issues in both hardware and software which can be summarized as follow.

For hardware:

(a) The robot did not walk in a straight path due motor imbalance and lack of feedback control.
(b) The Bluetooth module was having difficulty connected to an iPad when there are.
(c) The motor driver did not provide enough drive.
(d) The batteries did not last long.

For software:

(a) The commands were difficult to understand especially the "loop".
(b) The application could not select the Bluetooth because it was hard-wired to the software.
(c) The application did not know the status of the robot other than connected to Bluetooth.
(d) The application did not record the user information.
(e) The application did not keep track of the user answers and time required to solve each problem.

Thus we took the above issues into consideration when making an improvement to our existing platform from [7]. This section describes the technicality aspects of the design and implementation of the improved Arducation Bot as follows.

A. Hardware

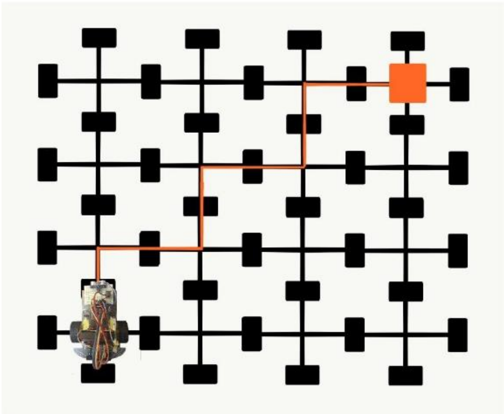


Fig. 1. The Arducation Bot with its playing board. The board is 4x4 grid allowing the robot to move along the lines.

The philosophy behind the creation of Arducation Bot, shown in Fig 1, is so that anyone can recreate the robot with off the shelf hardware. The components of the robot include: an Arduino UNO R3, two IR sensors, a sonar sensor, an L298n motor driver, two

DC motors, and an HM-10 Bluetooth BLE module. of the robot. The flowchart in Fig 3 also shows the

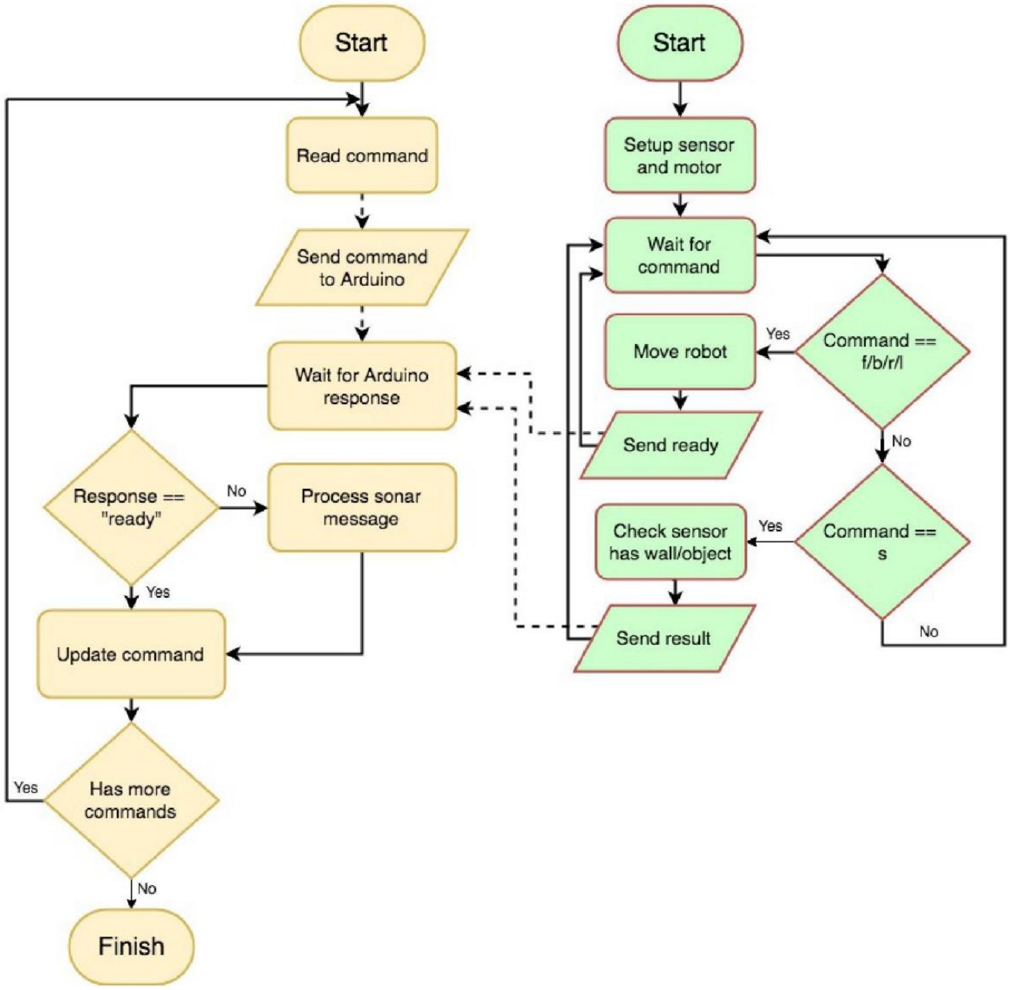


Fig. 3. Flowchart of Arduino (right) and iOS (left) programs.

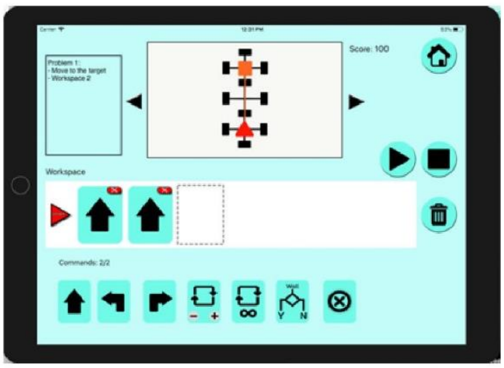
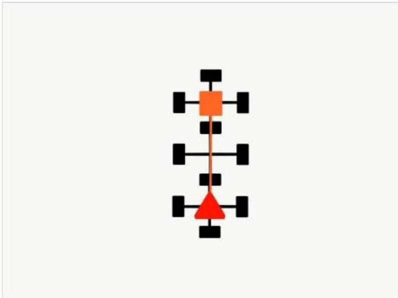
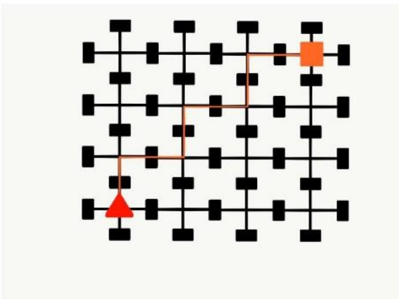


Fig. 4. The iOS Application for Arducation Bot platform.

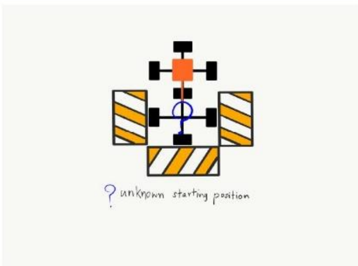
To understand if the Arducation Bot platform improves the students computational thinking skills, the time required for each group of students to finish each puzzle was recorded. This was the duration of when the puzzle first appeared on the screen up until when the students finished putting in all the commands they wanted in the workspace and hit run. As seen in Fig 7, the students gradually used less time to solve problems 1 through 4 (Puzzle01 - Puzzle04). More importantly, the deviation of time required to solve those problems become less varied. This shows a clear improvement in the ability to algorithmically solving sequencing problem. However, our assumption breaks down for Puzzle05 and Puzzle06 even though these two problems are not yet required more understanding of other concepts. As seen back in Fig **??** with all the puzzles, the answers for Puzzle05 and Puzzles06 require 7 and 11 commands respectively, thus skewing the results of the easy level puzzles.



(a) Example of an easy-level puzzle (Puzzle01)



(b) Example of a medium-level puzzle (Puzzle08)



(c) Example of a hard-level puzzle (Puzzle08)

Fig. 5. Web-application for water glider control and measurement data. The underwater glider can be controlled from the "Home" page through the "Controller" panel. The aerial and underwater measurements are displayed simultaneously on the "Survey" page

The medium level puzzles are Puzzle07 and Puzzle08. Both Fig 7 and Table I show that the students did not need a lot of time to understand the newly introduced concept of the loop after having gone through computational thinking exercise with previous problems. The time required to solve Puzzle07 is very

Fig. 6. Testing of Arducation Bot at the Thailand-Japan Student Science Fair2018. There are both Thai and Japanese students in is particular group of students.

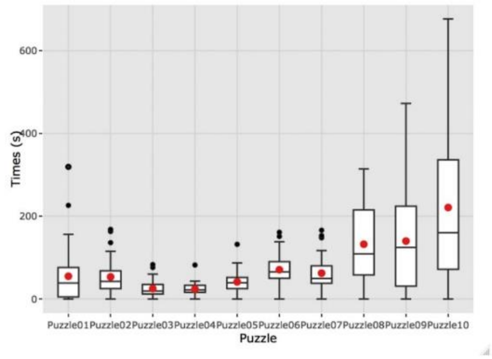


Fig. 7. Time required for each group of students to finish solving the puzzles.

similar to Puzzle02. However, the results from Puzzle08 presented a significant jump. The time required to solve the conditional problems in Puzzle09 and Puzzle10 also drastically differed from the rest of the puzzles. One explanation would be the students had not adequately been introduced to these two new programming concepts, therefore most of them were having difficulty understanding and solving these problems

TABLE I
STATISTICAL EVALUATION OF THE TIME REQUIRED TO SOLVE EACH PROBLEM FROM 180 STUDENTS

Problem	**min [s]**	**max [s]**	$\bar{x}$ **[s]**	σ **[s]**
Puzzle01	2	319	54.911	69.763
Puzzle02	8	168	53.500	45.664
Puzzle03	4	83	25.176	20.566
Puzzle04	12	82	24.029	15.526
Puzzle05	11	132	41.264	26.206
Puzzle06	36	161	70.588	42.648
Puzzle07	10	166	62.354	43.531
Puzzle08	15	314	132.176	92.605
Puzzle09	29	427	139.794	126.780
Puzzle10	28	676	220.788	206.140

V. Conclusion

This paper has demonstrated the feasibility of a combination of tangible and mobile technology to create a learning platform in computational thinking and computer science. The test results from Thai and Japanese students show the potential of the platforms effectiveness regardless of the users background. A clear pattern of improved computational thinking was shown in the collected data from the Arducation Bot. This platform presents a low-cost and easily accessible teaching medium that could potentially be beneficial for developing skills in computational thinking and computer science.

Acknowledgement

The authors would like to thank Princess Chulabhorn's College Phitsanulok, the Mobile Computing Lab at Naresuan University, and the Naresuan University Maker Club for their assistance.

References

[1] C. thinking for educators, "What is computational thinking," https://computationalthinkingcourse.withgoogle.com/unit?lesson=8&unit=1.

[2] J. M. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, Mar. 2006.

[3] E. M. Reeve, "Implementing science, technology, mathematics and engineering (stem) education in thailand and in asean," *Report Prepared for: The Institute for the Promotion of Teaching Science and Technology*, 2013.

[4] T. I. for the promotion of Teaching Science and Technology, "Ispt. new course "computational thinking" kills to prepare for the future," http://www.ipst.ac.th/index.php/news-and-announcements/articles/item/3285-2018-07-05-06-20-42.

[5] NECTEC, "Coding at school powered by kidbright with thailand 4.0," https://www.nectec.or.th/news/news-pr-news/kidbright-coding.html.

[6] M. U. Bers, "The tangiblek robotics program: Applied computational thinking for young children." *Early Childhood Research & Practice*, vol. 12, no. 2, p. n2, 2010.

[7] T. Phetsrikran, W. Massagram, and A. Harfield, "First steps in teaching computational thinking through mobile technology and robotics," *Asian International Journal of Social Sciences*, p. 37, 2014.

[8] S. Grover and R. Pea, "Computational thinking in k–12: A review of the state of the field," *Educational Researcher*, vol. 42, no. 1, pp. 38–43, 2013.

[9] C. Stephenson and V. Barr, "Defining computational thinking for k-12," *CSTA Voice*, vol. 7, no. 2, pp. 3–4, 2011.

[10] Wikipedia, "Computational thinking." https://en.wikipedia.org/wiki/Computational_thinking.

[11] C. at School Working Group, "Computer science: a curriculum for schools." http://tinyurl.com/y7bmeebo.

[12] BBC, "Introduction to computational thinking." https://www.bbc.co.uk/education/topics/z7tp34j.

[13] Barefoot, "computational thinking." https://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking/.

[14] F. B. V. Benitti, "Exploring the educational potential of robotics in schools: A systematic review," *Computers & Education*, vol. 58, no. 3, pp. 978–988, 2012.

[15] D. Alimisis, "Educational robotics: Open questions and new challenges," *Themes in Science and Technology Education*, vol. 6, no. 1, pp. 63–71, 2013.

[16] O. Mubin, C. J. Stevens, S. Shahid, A. Al Mahmud, and J.-J. Dong, "A review of the applicability of robots in education," *Journal of Technology in Education and Learning*, vol. 1, no. 209-0015, p. 13, 2013.

[17] C.-W. Chang, J.-H. Lee, P.-Y. Chao, C.-Y. Wang, and G.-D. Chen, "Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school." *Journal of Educational Technology & Society*, vol. 13, no. 2, 2010.

[18] M. U. Bers, L. Flannery, E. R. Kazakoff, and A. Sullivan, "Computational thinking and tinkering: Exploration of an early childhood robotics curriculum," *Computers & Education*, vol. 72, pp. 145–157, 2014.

[19] S. Hutamarn, S. Chookaew, C. Wongwatkit, S. Howimanporn, T. Tonggeod, and S. Panjan, "A stem robotics workshop to promote computational thinking process of pre-engineering students in thailand: Stemrobot."

[20] Code.org, "Hour of code: Moano." http://code.org/moana.

ICIC Express Letters
Part B: Applications
Volume **12**, Number **1**, January **2021**

ICIC International ©2021 ISSN 2185-2766
pp. 27–34

ARDUCATION BOT: COMPUTATIONAL THINKING COURSEWARE WITH IOS MOBILE APPLICATION AND EDUCATIONAL ROBOTICS

Titipan Phetsrikran[1], Antony Harfield[1], Supiya Charoensiriwath[2] and Wansuree Massagram[1,*]

[1]Department of Computer Science and Information Technology
Faculty of Science
Naresuan University
Phitsanulok 65000, Thailand
{ titipanp60; antonyh }@nu.ac.th; *Corresponding author: wansureem@nu.ac.th

[2]National Electronics and Computer Technology Center
Patumtani 12120, Thailand
supiya.charoensiriwath@nectec.or.th

Received May 2020; accepted August 2020

Abstract. *The purpose of this project is to develop an educational platform for improving primary students' computational thinking ability. Arducation Bot combines tangible technology and mobile technology to create intuitively approachable teaching computational thinking. The result of the Arducation Bot was tested with 177 primary school students from Thailand. A clear pattern of improved computational thinking was demonstrated by the pre-test and post-test scores and related data from the Arducation Bot. This project presents a low-cost and intuitive teaching tool that can effectively develop skills in computational thinking and prepare students for computer science.*
Keywords: Computational thinking, Educational robotics, Tangible technology, Mobile technology

1. **Introduction.** The term "computational thinking" refers to a thought process to develop problem-solving skills that breaks down any task into smaller parts, finding patterns in each problem, and then logically presenting solutions using algorithms that can be repeated and followed. Computational thinking can help solve problems not only in computer science and mathematics but also in everyday life. Given its current relevance and importance, there is significant demand in Thailand and internationally for computational thinking in schools starting at a young age [1-3].

The purpose of this work is to provide a way to increase computational thinking skills by combining tangible and mobile technologies to develop a platform that is both accessible and effective. The iOS mobile application delivers challenging puzzles, which are divided into four units. Each unit teaches an important concept of computational thinking: sequences, loops, conditions, and conditions with loops. The Arduino-based robot car provides a tangible medium for interaction – young students can see the results of their thought and programming in the real world.

Tangible technology has proven to be an effective tool for children's mental development – mapping concrete objects to abstract reasoning such as computational thinking [4]. Existing computational thinking study curriculums such as code.org or MIT Media Lab's Scratch programming language often focus on creating codes on a personal computer or laptop. Using educational robotics such as Ozobot, Lego Mindstorms, or Arducation Bot offers the benefit of tangible technology and encourages increased collaboration among children. To date there have been two initial versions [5,6] of the Arducation Bot. The

DOI: 10.24507/icicelb.12.01.27

purpose of the current study is to further improve the system and its integrated courseware to teach the target computational thinking skills even more effectively and provide quantitative measurement of student improvement.

The organization of this paper is as follows. The background of related domains is described in Section 2. The methodology of how to construct the hardware, software, and courseware is illustrated in Section 3. Section 4 depicts the experimental procedures, their results, and the discussion of this research's significances. The conclusion and direction for future work for this study are summarized in Section 5.

2. **Literature Review.** The literature can be categorized into three main project-related headings: computational thinking, educational robotics, and courseware.

2.1. **Computational thinking.** The definition of computational thinking is a basic process for solving problems. There is a six-key concept of computational thinking, with five approaches [7,8]: (a) "logic" is reasoning that helps us explain why something happens, (b) "algorithms" are a sequence of instructions to solve problems, (c) "decomposition" is a process of breaking down a task into smaller pieces, (d) "patterns" are identifying details, creating rules and solving more general problems, (e) "abstracting" is simplifying or identifying something important without worrying about the details, and (f) "evaluation" is about making estimates of an objective in a systematic way. These concepts can improve the development of the five approaches: 1) "tinkering" is often to try something new to discover how it works, 2) "creating" is about making and planning something, 3) "debugging" is finding and fixing errors in code or algorithms, 4) "persevering" is a never give up attitude even though the problem is hard, and 5) "collaboration" is people working together to develop a good environment.

2.2. **Educational robotics.** The field of educational robotics includes many different facets, such as physical platforms, educational resources, and tangible technology. Robotics are commonly used in educational activities to transfer academic knowledge and skills related to Science, Technology, Engineering, and Mathematics (STEM) [9]. Educational robotics provides a tangible way that students can easily send instructions to a robot and have their input validation without yet having to learn syntax [10].

Active, cooperative, and problem-based learning using educational robotics and mobile technology are suitable for both undergraduate and graduate robotics education [11,12]. Another study shows the impact of educational robotics on children's technical, social, and science-related skills. The study depends on a two-point measurement (pre and post-test) to evaluate the impact, with each measurement a multiple-choice questionnaire [13].

Chang et al. believed educational robotics can help the kids in developing collaboration and communication, problem-solving abilities, critical thinking skills, and creativity among students [14]. Furthermore, teachers could completely include educational robotics in the young children's computer programming curriculum because the robots offer a better tinkering approach than computer monitors [15]. Thus, educational robotics are appropriate and have been practical to students of different age groups.

The arrival of educational robot use in schools has had a significant impact. For example, Nugent et al. used the robot together with geography technology to teach students about science, technology, engineering, and mathematics (STEM) [16]. Another study by Alimisis used educational robotics to identify new trends and challenges that focus on using robotics as a tool for creativity and other 21st century skills [17]. Williams et al. studied to estimate the impact of educational robotics on high school students' physics knowledge and scientific investigation skills [18]. Chin et al. used educational robotics to develop a system that provides an attractive teaching application about multimedia objects and its effect on student performance and motivation [19]. By no means educational robotics are the silver bullet. The tools are only as effective as the study plans and teaching materials.

2.3. **Courseware.** It is a term generally used to describe educational materials. These materials could be a kit to teach, train, or tutor the students. Most courseware is associated with technology-based materials. The term "courseware" is commonly referred to training for personal computers, software packages, or IT certification programs [20,21]. The followings are common courseware materials: instructor-led video or notes, self-directed computer-based training (CBT), interactive tutorials, and live or webinar.

The courseware in [22] demonstrates how STEM-driven computer science education supports the development of computational thinking at the high school. Well-known examples for computer science and computational thinking are code.org and ScratchEd [23,24] which use web delivered interfaces through which children learn how to write code. While these exercises offer a wide range of puzzles, the programming portion requires students to sit in front of their computers. This study wants to encourage the five computational thinking approaches: tinkering, creating, debugging, persevering, and collaboration. Playing with a tangible and mobile device in a group should encourage all five approaches, especially collaboration.

3. **Methodology.** The combination of tangible and mobile technologies created particularly for this project is called "Arducation Bot" [5,6]. Arducation Bot was designed to be an educational platform for improving primary school students' computational thinking ability. The initial version of the platform was found to have a few difficulties in both hardware and software. In hardware, the robot was not configurable due to all parameters being hard-coded and the line-following sensor did not track as well as needed. In software, the application UI was difficult to understand. This section describes the technical aspects of Arducation Bot and more generally, project implementation.

3.1. **Hardware.** The components of the robot include an Arduino UNO R3, three IR sensors, an ultrasonic sensor, an L298n motor driver, two DC motors, an HM-10 Bluetooth BLE module, and a 3.7V li-ion battery. Everything is placed on a plastic frame, with the motor shafts serving as axels for the wheels. The circuit diagram is shown in Figure 1. The Arduino provides control and processing power for the robot. The two types of sensors, ultrasonic and IR, provide input feedback. The motors, controlled by the L298, provide physical actuation. Bluetooth provides communication between the robot and the iOS application.

3.2. **Software.** Two kinds of software have been developed for Arducation Bot: Arduino and Swift. The two main functions are to control the robot and to communicate with an iPad. The Swift-based iOS application provides an interface between the user and the robot. The user's commands are sent from an iPad to the Arduino board through Bluetooth communication. The communication flow starts by the robot sending a "ready" message to the iPad indicating the robot's readiness to receive a command. The iPad sequentially sends commands to the robot, one at a time. After the robot finishes processing each individual command, it will send another "ready" message in order to receive the next command. Some commands are related to movement and others are related to obtaining information from the robot. For example, informational commands ask the robot if there is an obstacle in front of the robot. The puzzles and the interface in the iOS application styled after the curriculum and online sessions used by code.org. The target users of this platform are pupils above seven years of age. The UI/UX was designed to be intuitive and interactive, with users dragging and dropping the commands into the workspace as shown in Figure 2.

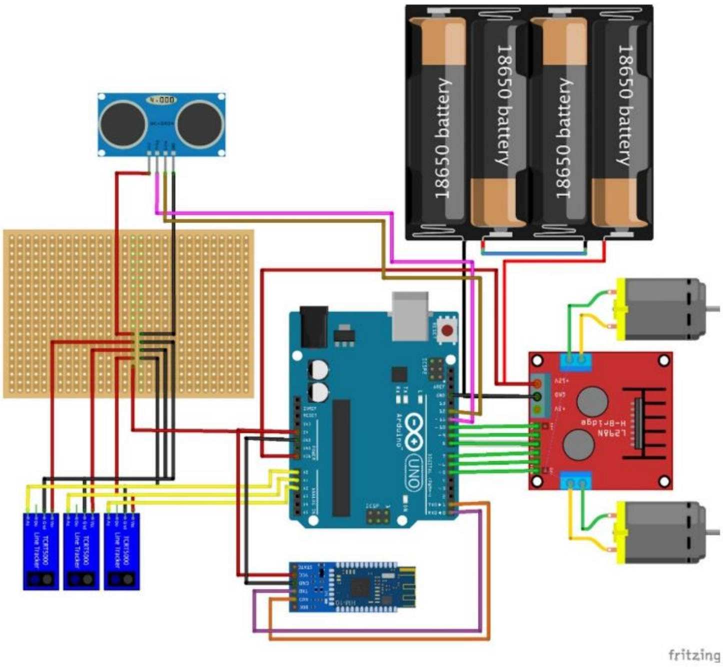


FIGURE 1. The Arducation Bot circuit diagram. Its components include an Arduino UNO R3, three IR sensors, an ultrasonic sensor, an L298n motor driver, two DC motors, and an HM-10 Bluetooth BLE module.

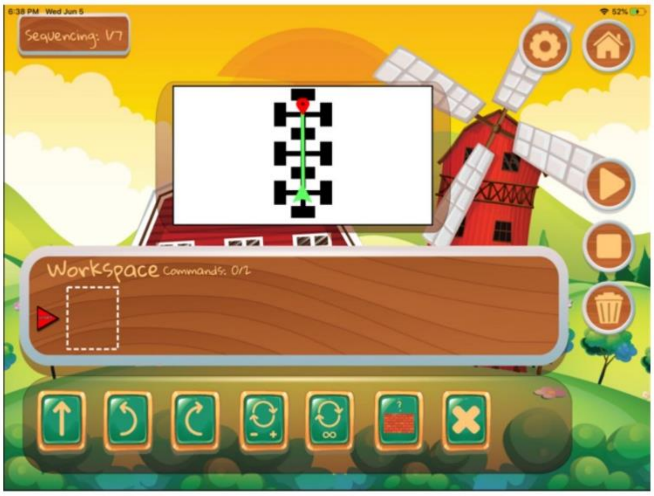


FIGURE 2. iOS application

The application starts the children off with a simple puzzle that explains basic commands to control the robot. The application contains twenty different puzzles, divided into four units of increasing complexity. The first unit requires the children to understand sequencing. The second unit teaches the concept of loops. The third unit is about conditionals. The fourth unit is a combination of a loop and a conditional. The design of the puzzles is explained in the courseware section below.

3.3. **Courseware.** Arducation Bot is modeled on concepts and designs shared by many online courses with block programming. The Arducation Bot platform consists of four units, each covering one topic with a series of target concepts, and the teaching of each target concept is built around one puzzle. The four units are Sequencing, Loops, Conditions, and Conditions with loops. In Unit 1 (Sequencing), children learn the concept of sequencing an activity into steps. In this unit, they also learn the basic commands and movements of the robot. In Unit 2 (Loops) children learn the concept of using a loop to repeat a procedure in order to accomplish a goal. In Unit 3 (Conditions) children learn the concept of expressing instructions based on a condition, using "If" and "Then". For example, "IF you meet an obstacle THEN turn left, but if you do not meet an obstacle, then continue straight". In this unit, the children learn how to divide a big problem into a sequence of smaller tasks. In the above example, there are three such smaller tasks: 1) check if there is an object, 2) if there is an object turn left, and 3) if there is no object, continue forward. Finally, in Unit 4 (Conditions with loops) children learn to combine the two concepts of If Conditions and Loops in order to reach a goal.

Children must learn various commands and think about the algorithm to solve these puzzles because there is a limit number of commands in each puzzle. For example, the first puzzle requires only two commands to solve. The loop concept is introduced with a puzzle that uses commands. Another concept that Arducation Bot uses to design and improve children's skills is the conditional puzzle. This puzzle uses the conditions command when meeting the obstacles then stop. Three examples of puzzles shown in Figure 3 illustrate the different themes. The design of all puzzles incrementally reveals the children's computational thinking concept as explained in the literature review section.

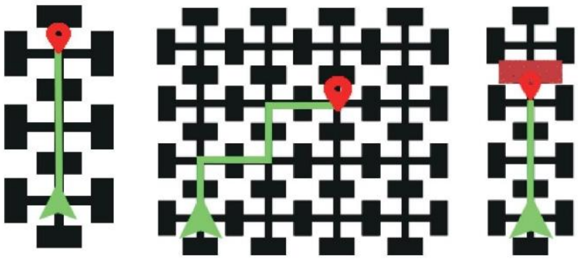


FIGURE 3. Examples of Arducation Bot: (left) the first puzzle, (middle) loop puzzle, and (right) condition puzzle

4. **Result and Discussion.** The Arducation Bot was tested in June 2019 with students from various primary schools (mostly around Phitsanulok, a provincial city in Thailand) during a one-day event called Computational Thinking for Kids, which was held four times, twice at Naresuan University and twice at St. Nicholas School in Phitsanulok. Each day, the students were split into ten groups, with four or five students in each group. Computational Thinking for Kids had two main sessions each day, called Unplugged (in the morning) and Arducation Bot (in the afternoon). At the beginning of the day, each student took a pre-test, and at the end of the day, they took a post-test. Data was collected from these two tests and processed to evaluate the difference in the computational thinking skills of each student before and after participating in the one-day event. The results are shown in Table 1.

This data was derived from a total of 177 students participating over the course of the four days. It is known that students at Events 2 and 3 had never studied computational thinking before, but Events 1 and 4 may have included some students who had studied

TABLE 1. Testing events and statistical analysis of the pre-test and post-test out of 10 points where $\bar{x}$ is the students' average score for each age group and σ is a standard deviation of the test scores

Event No.	**Participating students (Age)**	**Students count**	**Pre-test score**		**Post-test score**	
			$\bar{x}$	σ	$\bar{x}$	σ
1	Grade 1-6 (6-12)	46	5.64	2.72	7.25	2.16
2	Grade 2-3 (7-8)	50	4.7	1.19	6.95	1.82
3	Grade 4-5 (9-10)	50	3.62	1.83	5.98	2.54
4	Grade 1-6 (6-12)	31	5.48	2.82	7.6	1.69
		177	**4.86**	**2.14**	**6.95**	**2.05**

computational thinking previously. Looking at the mean points from the students' pre-tests and post-tests, the post-test scores were clearly improved, with a 43 percent increase over the pre-test scores. Standard deviations were 2.14 for the pre-test and 2.05 for the post-test.

The time required by each student to solve each puzzle was recorded in the iOS application. The puzzles are divided into four units: 1) Sequencing, 2) Loops, 3) Conditions, and 4) Conditions with loops. By solving these puzzles, the student should obtain computational thinking skills. The skills are in logic, decomposition, algorithms, abstraction, patterns, and evaluation. Each of the twenty puzzles requires the student to figure out one or more correct algorithms to move the robot from a starting point to a finishing point. When the student thinks they have figured out the correct algorithm(s) of the puzzle they are working on, they push the Run button. Then the robot will move according to their instructions (algorithms). However, if their algorithms are wrong, the student is informed and asked to try again. In order to understand how the Arducation Bot platform improves a student's computational thinking skills, the time required by each student to correctly answer each puzzle was automatically recorded. Figure 4 summarizes the average time (of all 177 students) spent correctly answering each of the 20 puzzles.

Interestingly, Units 1, 3, and 4 follow a similar pattern in Figure 4. In each of these three units, the student gradually used relatively more time per question at the start of the unit and relatively less time per question at the end of the unit. Only Unit 2

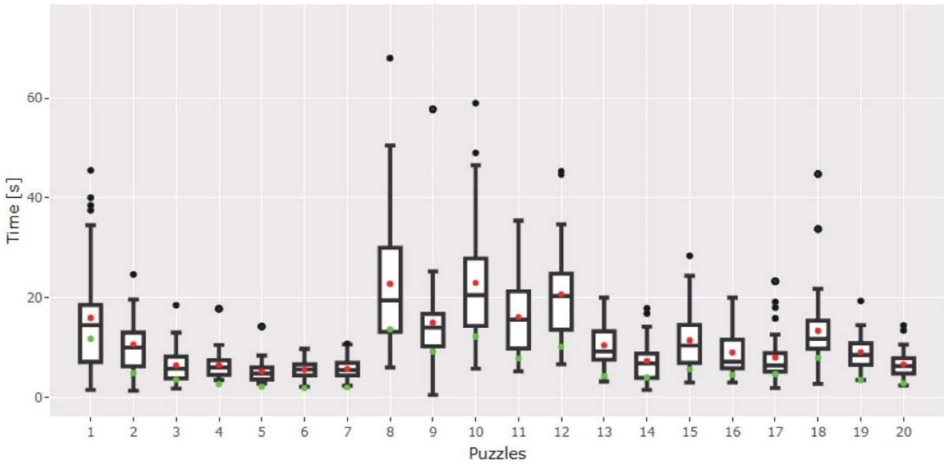


FIGURE 4. The average time required to successfully finish each puzzle

did not show this pattern. Since all the questions within one unit were approximately equally difficult, the shorter answering times as the unit progresses seems to show a clear improvement in the student's ability to understand the concepts in that unit. However, it is unclear why unit 2 did not follow the same pattern.

It should also be pointed out that conditions varied on four different test days. During the two test sessions at St. Nicholas School, students were on their home turf so to speak and being supervised by their regular teachers. Therefore, the students tend to be relatively well behaved. On the other hand, during the two other test sessions at Naresuan University, students were visiting the campus to attend and were surrounded by a new environment full of novel stimuli; thus they tended to be relatively less behaved and less focused on the task at hand. However, when the test results from the two locations are separated and compared, the results are strikingly similar to each other, as seen below in Figure 5. This similarity of outcome even in two different settings and atmospheres suggests that the test is valid and meaningful. The Arducation Bot platform successfully improved the students' computational thinking skills, enabling them to better understand and solve various computational tasks.

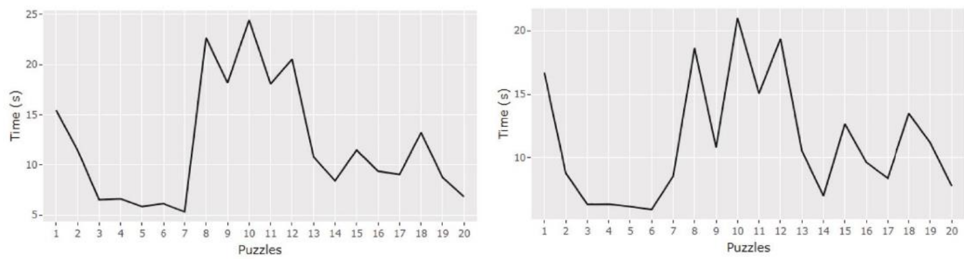


FIGURE 5. The average time required to successfully finish each puzzle at two different locations: (left) at the student's own school and (right) at Naresuan University

5. **Conclusions.** This study developed a tangible tool that utilizes mobile technology to create an educational platform in computational thinking. The results from 177 primary school students who participated in the Computational Thinking for Kids event have shown the potential of this courseware platform. A clear pattern of improved computational thinking was demonstrated by the pre-test and post-test scores and related data from the Arducation Bot. This platform presents a low-cost and intuitive teaching tool that can effectively develop skills in computational thinking and prepare students for computational thinking and computer science skills. To further prove the effectiveness of this proposed study, a comparison with extant pedagogical programs for computational thinking is needed. Furthermore, to broaden the impact of this study, the researchers are planning these two parallel efforts: the open-source hardware distribution for low-cost and the integration into classroom curriculums.

Acknowledgment. This study was made possible by a generous grant from the Thailand Graduate Institute of Science and Technology (TGIST) fund (Grant No. SCA-CO-2560-4530-TH). The authors would also like to thank the NU Maker Club and the Mobile Computing Laboratory at Naresuan University for their assistance.

REFERENCES

[1] E. M. Reeve, *Implementing Science, Technology, Mathematics and Engineering (STEM) Education in Thailand and in ASEAN*, Report Prepared for: The Institute for the Promotion of Teaching Science and Technology, 2013.

[2] Office of the Education Council, *The National Scheme of Education B.E. 2560-2579 (2017-2036)*, 1st Edition, Prigwhan Graphic, Bangkok, 2017.
[3] NECTEC, *KidBright*, https://www.nectec.or.th/news/news-pr-news/kidbright-coding.html/, Accessed on August 20, 2018.
[4] C. O'Malley and D. S. Fraser, Report 12: Literature review in learning with tangible technologies, *NESTA Futurelab*, 2004.
[5] T. Phetsrikran, W. Massagram and A. Harfield, First steps in teaching computational thinking through mobile technology and robotics, *Asian International Journal of Social Sciences*, p.37, 2014.
[6] T. Phetsrikran, W. Massagram, T. Phoka and A. Harfield, A feasibility study of Arducation Bot: An educational robotics and mobile application kit for computational thinking skills, *The 22nd International Computer Science and Engineering Conference (ICSEC)*, pp.1-6, 2018.
[7] Barefoot, *Computational Thinking*, https://www.barefootcomputing.org/concept-approaches/computational-thinking-concepts-and-approaches/, Accessed on September 18, 2017.
[8] BBC, *Introduction to Computational Thinking*, https://www.bbc.co.uk/bitesize/topics/z7tp34j, Accessed on September 19, 2017.
[9] S. Bargagna, E. Castro, F. Cecchi, G. Cioni, P. Dario, M. Dell'Omo and G. Sgandurra, Educational robotics in down syndrome: A feasibility study, *Technology, Knowledge and Learning*, pp.1-9, 2018.
[10] J. Shim, D. Kwon and W. Lee, The effects of a robot game environment on computer programming education for elementary school students, *IEEE Trans. Education*, vol.60, no.2, pp.164-172, 2017.
[11] G. Cielniak, N. Bellotto and T. Duckett, Integrating mobile robotics and vision with undergraduate computer science, *IEEE Trans. Education*, vol.56, no.1, pp.48-53, 2013.
[12] L. D. Riek, Embodied computation: An active-learning approach to mobile robotics education, *IEEE Trans. Education*, vol.56, no.1, pp.67-72, 2013.
[13] M. Kandlhofer and G. Steinbauer, Evaluating the impact of educational robotics on pupils' technical- and social-skills and science-related attitudes, *Robotics and Autonomous Systems*, vol.75, pp.679-685, 2016.
[14] C. W. Chang, J. H. Lee, P. Y. Chao, C. Y. Wang and G. D. Chen, Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school, *Journal of Educational Technology & Society*, vol.13, no.2, pp.13-24, 2010.
[15] M. U. Bers, L. Flannery, E. R. Kazakoff and A. Sullivan, Computational thinking and tinkering: Exploration of an early childhood robotics curriculum, *Computers & Education*, vol.72, pp.145-157, 2014.
[16] G. Nugent, B. Barker, N. Grandgenett and V. I. Adamchuk, Impact of robotics and geospatial technology interventions on youth STEM learning and attitudes, *Journal of Research on Technology in Education*, vol.42, no.4, pp.391-408, 2010.
[17] D. Alimisis, Educational robotics: Open questions and new challenges, *Themes in Science and Technology Education*, vol.6, no.1, pp.63-71, 2013.
[18] D. C. Williams, Y. Ma, L. Prejean, M. J. Ford and G. Lai, Acquisition of physics content knowledge and scientific inquiry skills in a robotics summer camp, *Journal of research on Technology in Education*, vol.40, no.2, pp.201-216, 2007.
[19] K. Y. Chin, Z. W. Hong and Y. L. Chen, Impact of using an educational robot-based learning system on students' motivation in elementary education, *IEEE Trans. Learning Technologies*, vol.7, no.4, pp.333-345, 2014.
[20] Idea Group. Lnc., *What is Courseware*, https://www.igi-global.com/dictionary/disruptive-technologies-and-education/6098/, Accessed on August 21, 2018.
[21] Margaret Rouse, *Courseware*, https://whatis.techtarget.com/definition/courseware/, Accessed on August 21, 2018.
[22] R. Burbaitė, V. Drąsutė and V. Štuikys, Integration of computational thinking skills in STEM-driven computer science education, *Global Engineering Education Conference (EDUCON)*, pp.1824-1832, 2018.
[23] F. Kalelioğlu, A new way of teaching programming skills to K-12 students: Code.org, *Computers in Human Behavior*, vol.52, pp.200-210, 2015.
[24] K. Brennan, *ScratchEd*, https://scratched.gse.harvard.edu/, Accessed on October 28, 2019.