Let's Crack Let's Code

with KidBright

English Version

STUDENT HANDBOOK

Let's Crack Let's Code with KidBright

KidBright

STUDENT HANDBOOK

Sponsored by

Ministry of Higher Education
Science Research and Innovation

NSTDA

NECTEC
a member of NSTDA

Let's Crack Let's Code with KidBright

STUDENT HANDBOOK

National Electronics and Computer Center

National Science and Technology Development Agency

Let's Crack Let's Code with KidBright /by Embedded System Technology Research Team, National Electronics and Computer Technology Center, National Science and Technology Development Agency. -- Pathum Thani : Thammasat University Press, 2019.

200 pages : colored illustrations

1. Computers 2. Digital communications 3. Data transmission systems 4. Multimedia communications 5. Computer network protocols 6. Computer algorithms

I. National Science and Technology Development Agency II. National Electronics and Computer Technology Center III. Embedded System Technology Research Team IV. Title

TK5105 004.6

Published by Ministry of Higher Education Science Research and Innovation

National Electronics and Computer Center

National Science and Technology Development Agency

Ministry of Science and Technology, Thailand

112 Phahonyothin Road, Khlong Nueng, Khlong Luang District, Pathumthani 12120, Thailand.

Tel: 662-564-6900 Fax: 662-564-6901-3 NECTEC Call Center: 662-524-9200 e-mail: info@nectec.or.rh

Contents

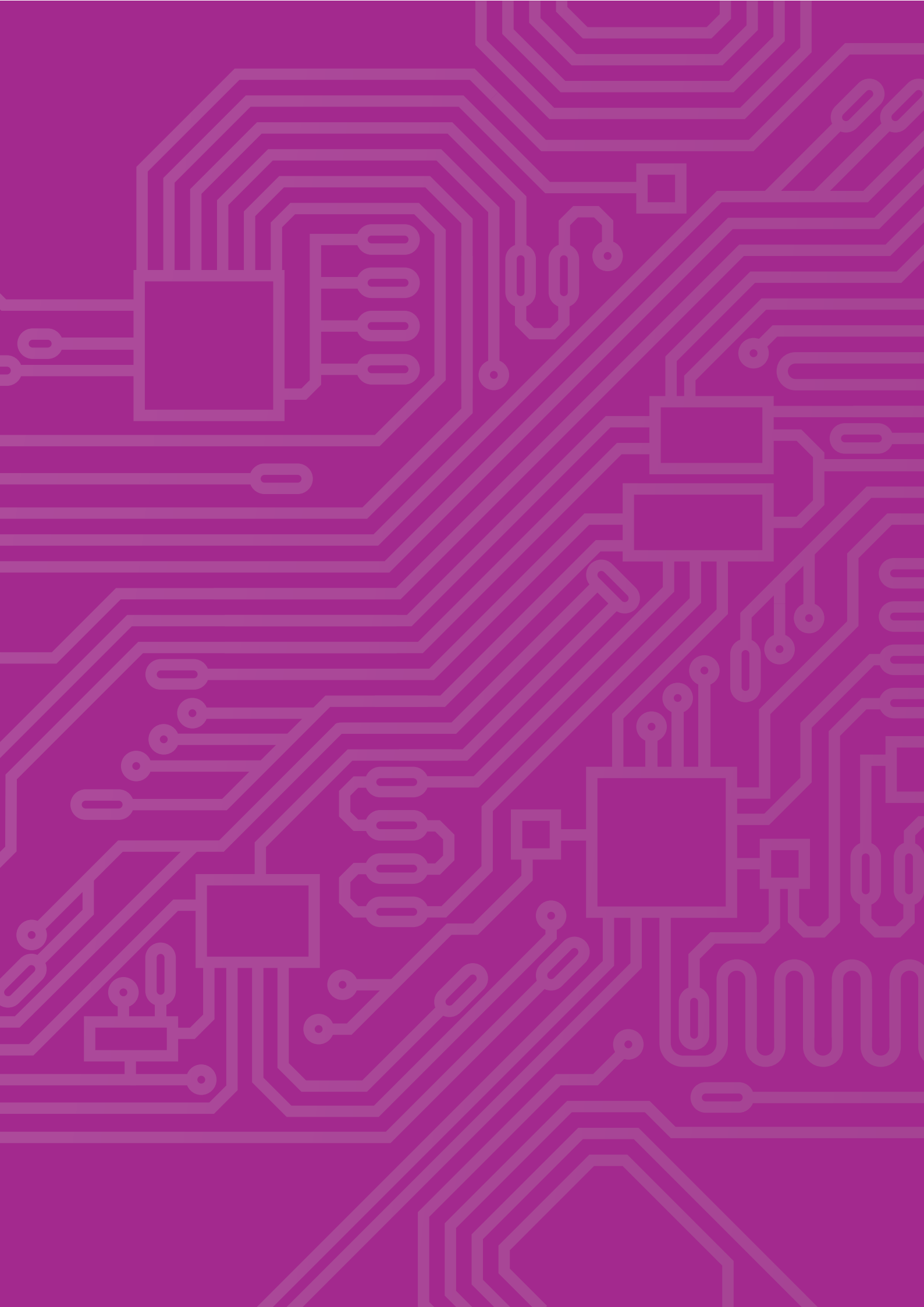
Chapter 1

Get to Know the Computational Thinking

Learning Objectives

At the end of the learning process in this chapter, students should be able to:

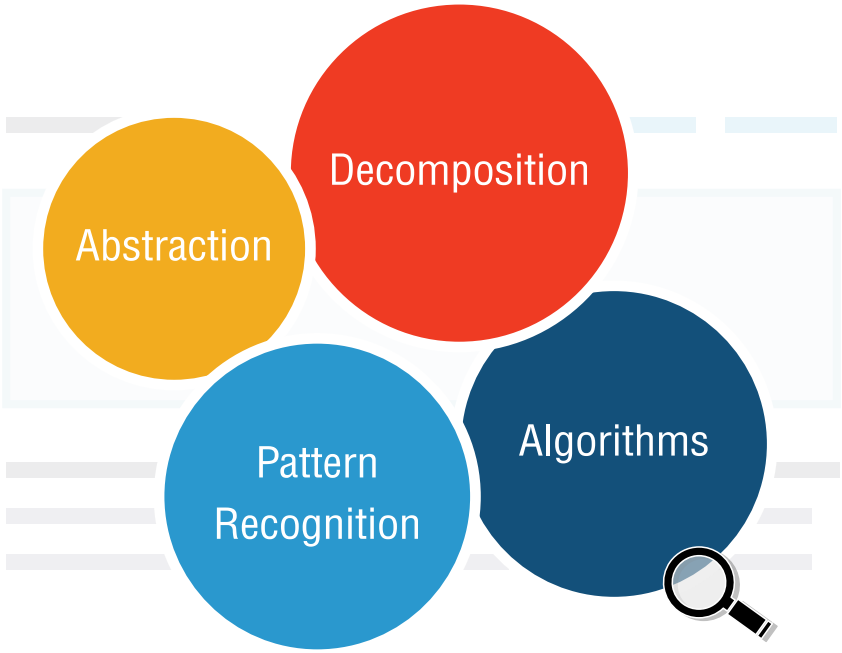
1. Explain the concept of computational thinking.
2. Demonstrate skills necessary to solve problems by computational thinking including:
   a. Decompose a complicated problem into simpler ones.
   b. Recognize patterns among problems.
   c. Design algorithms to solve problems.



Essence of computational thinking

Computational thinking is an important analytical skill for solving problems in a systematic manner.

Computational thinking consists of four principal components that make up the analytical process.



Decomposition
Decomposition is a process to decompose, or break down a complex system into subsystems that we can analyze easily.

Abstraction
Abstraction is a process to filter out irrelevant information and keep only the important part.

Algorithms
Algorithms are a step by step processes or sets of instructions for solving a problem.

Pattern Recognition
Pattern recognition is a process to identify and recognize the similarites (patterns) among or with in problems.

Activity

In the following activities, students will study the principal components of computational thinking. Questions asked in each case correspond to the computational thinking components.

Activity 1.1 Abstraction

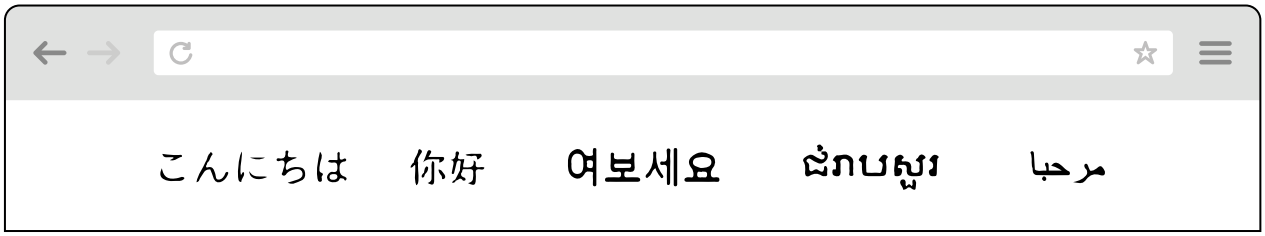
> "Abstraction is a process to filter out irrelevant information and keep only the important part."

Look at the following figure.

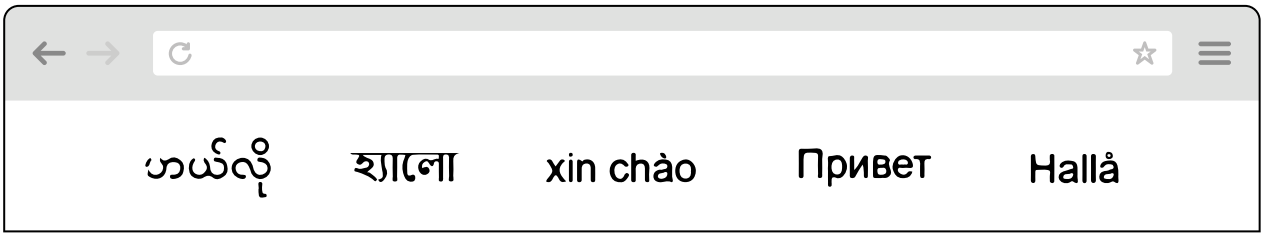
Figure 1.1 Words in different languages.
Source: http://translate.google.co.th/

Question 1: All these words in different languages in Figure 1.1 have the same meaning. What is the meaning of each word?

Look at the first line.



Can you answer the question? If you cannot, move to the second line.



Do you know the meaning of words in this line? If you don't, you may need to read the third line.

When you see the word "Hello" in this line, you may conclude that all wards in Figure 1.1 are words that are written in different languages and have the same meaning.

Finding the most important data is a challenging task because it depends on a person's prior knowledge and pervious experiences.

Activity 1.2 Algorithms

> Algorithms are a step by step processes or sets of instructions for solving a problem.

Consider the following situation.

A group of 30 students wants to do business. They plan to buy bread loaves from a supermarket at the cost of 55 cents each. They will then slice and pack the bread into smaller packages to sell in the school.

Question 2: What is the procedure used by the students to do the bread business?

The students have a process, or an algorithm, for doing their business that may have the following steps: forming a group, brainstorming, starting the bread shop, raising funds, buying supplies, and selling their product. These steps are illustrated in Diagram 1.1.

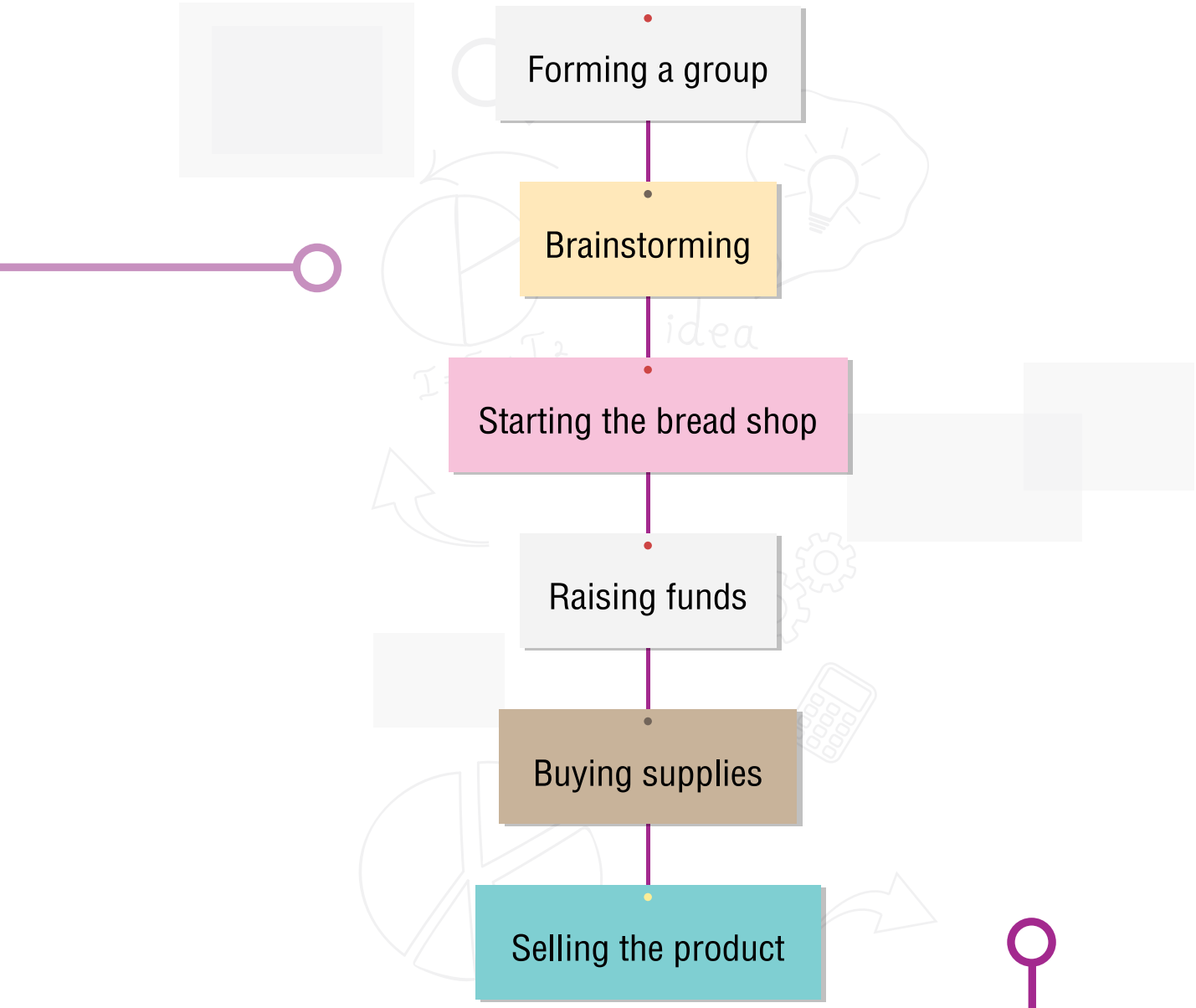


Diagram 1.1 Steps for doing the bread business.

Question 3: Review the business procedure and answer the question, how can we control the product quality?

Students have to brainstorm for setting up a business processes. For example, they raise funds by collecting 10 cents from each member of the group and use the money to buy three big loaves (55 cents each). They then slice and pack the bread into small, but beautiful packages. In total, they make 70 packages and set the price at 7 cents each. These students record income and expenses as shown in Table 1.1.

Table 1.1 Income-expense account of the bread business.

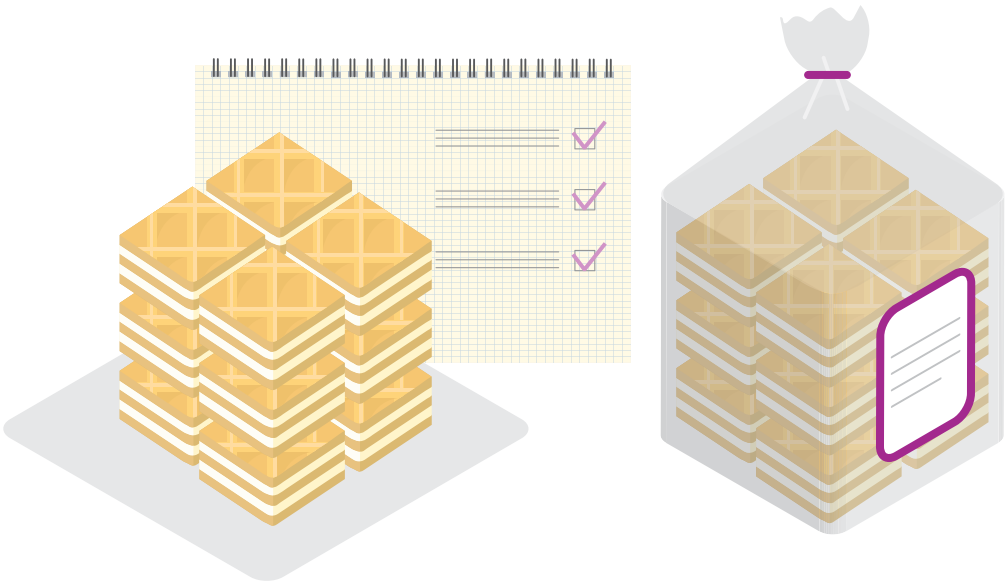
Date	Description	Income (cents)	Expense (cents)
19 May 2018	Collect 10 cents from each member	10x30 = 300	-
20 May 2018	Buy 3 big bread loaves (55 cents each)	-	165
	Buy plastic packages	-	25
	Buy a wrapping paper	-	10
21 May 2018	Sell 33 packages (7 cents each)	7x33 = 231	-
22 May 2018	Sell 21 packages (7 cents each)	7x21 = 147	-
23 May 2018	Sell 16 packages (7 cents each)	7x16 = 112	-

Source: Adapted from the teacher guideline for the learning area of occupations and technology, the 6th grade (Thai Ministry of Education).

The students have to control the quality of each package of sliced bread to meet certain criteria, e.g. each package must be clean, beautiful, and sealed well, and the total weight of each package should be the same. The packaging process is detailed on Diagram 1.2.

To answer Question 2, we draw a simple sequential procedure, as shown in Diagram 1.1. To answer Question 3, however, we need to consider the more complicate process of packaging which requires conditional steps and iterations as shown in Diagram 1.2.

A sequential process consists of steps for systematically solving a problem from the beginning to the end in one direction or in a sequential order. For a conditional process, some conditions must be checked, e.g. in the bread business, the weight and the appearance of the bread slice package must be verified to meet the required criteria. If the condition is false, the step is repeated until the condition turns true. For example, if the weight of a package is not correct, we have to add or remove pieces of bread. After obtaining the right weight, we proceed to the next step. Starting from the first package, we also check a condition whether we have all 70 packages done or not. If not, we proceed with the same procedure until we get all 70 packages.



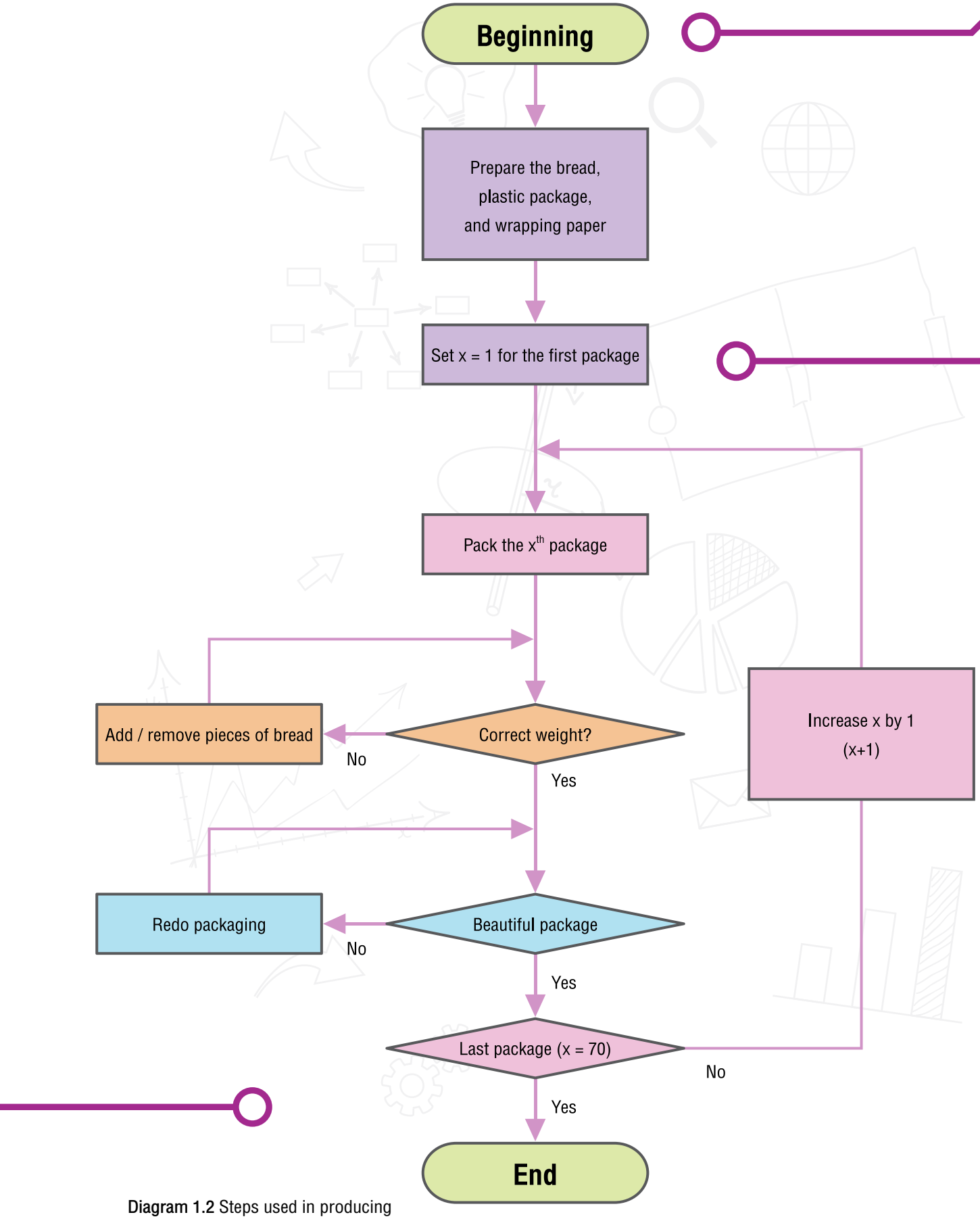


Diagram 1.2 Steps used in producing the product of the bread shop.

Question 4: Can students make a profit from the bread shop business?

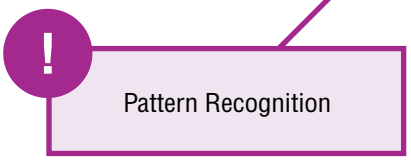
You can decompose activities of the bread shop into sub-activities, as shown in the following diagram.

Diagram 1.3 Sub-activities of the bread shop.

Students can use the income-expense account in Table 1.1 to calculate the daily balance as shown in Table 1.2, in order to answer Question 4.

Table 1.2 Daily balance of the bread shop.

Date	Description	Income (cents)	Expense (cents)	Balance (cents)
19 May 2018	Collect 10 cents from each member	300	-	300
20 May 2018	Buy 3 big bread loaves (55 cents each)	-	165	300 - 165 = 135
	Buy plastic packages	-	25	135 - 25 = 110
	Buy a wrapping paper	-	10	110 - 10 = 100
21 May 2018	Sell 33 packages (7 cents each)	231	-	100 + 231 = 331
22 May 2018	Sell 21 packages (7 cents each)	147	-	331 + 147 = 478
23 May 2018	Sell 16 packages (7 cents each)	112	-	478 + 112 = 590
	Total	300 + 490	200	



The daily balance can be calculated by considering the three factors: the previous balance, the daily income and the daily expense. The mathematical equation which expresses the relationship among the three factors is:

Abstraction

(1.1) Current balance = Previous balance + Current income - Current expense

On May 23, 2018, the day that the products are sold out, the balance is 590 cents. Since the total cost is 200 cents, the total income from the sales is 490 cents, and the money invested is 300 cents, the profit then equal to.

(1.2) 490 - 200 = 290 baht.

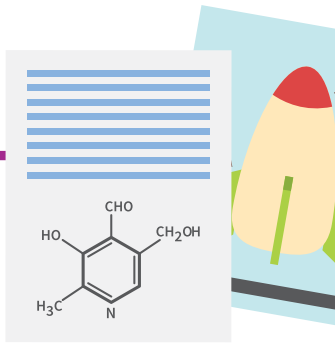
We conclude that the bread shop earns a profit of 290 baht.

It is important to ask correct questions in solving a problem. We can also ask more questions about this bread shop business including:

Question 5: How do these students adapt the proposed business processes in order to increase the profit?

Question 6: When does the break-even point occur? (Note that the break-even point is the point at which the total income and the total cost are equal).

Exercise



Exercise 1

The 4x100-meter relay race is an athletics event where a team of four runners complete 100 meters each. The relay race requires certain knowledge, skills and unity among the team members. They need to understand the rules or the race and have the necessary skills for proper warming-up, holding and passing the baton, positioning of the runners, etc.

Decompose the activities of a 4x100-meter relay race team and identify the duties of each runner. Prepare appropriate practice schedules for each of them.

Exercise 2

Consider the following triangles.

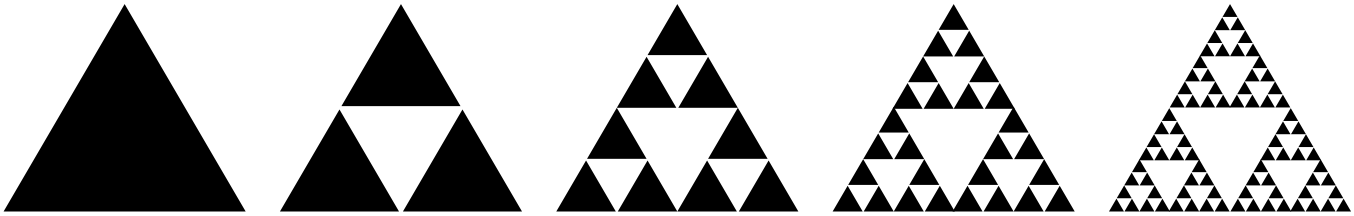


Figure 1.2 Triangles.

The triangles in Figure 1.2 (a) is an equilateral triangle where the length of each side is equal to x and the area is y. Perform the following tasks with proper mathematical equations.

1. Determine the side length of the black triangles and that of the white triangle in Figure 1.2 (b).

2. Determine the area of the black triangles and that of the white triangle in Figure 1.2 (b).

3. Determine the side lengths of the black triangles and those of the white triangles in Figure 1.2 (c), Figure 1.2 (d), and Figure 1.2 (e).

4. Determine the areas of the black triangles and those of the white triangles in Figure 1.2 (c), Figure 1.2 (d), and Figure 1.2 (e).

5. Find the relationship between the side lengths and the areas among these triangles.

6. Assume that there be Figure 1.2 (f) following the same pattern. What would it look like?

Exercise 3

Consider animals in the following table.

Elephant	Giraffe	Zebra	Rhinoceros	Buffalo
Cat	Tiger	Lion	Panther	Cheetah
Pig	Chipmunk	Beaver	Squirrel	Porcupine
Walrus	Seal	Dolphin	Whale	Dugong
Gibbon	Chimpanzee	Loris	Gorilla	Human

Answer the following questions.

1. What are the common features of five species in each row?

2. What are the features common to all the species in the table?

3. What features do these 25 animals have in common with the bat and the marmot?

Exercise 4

Global handwashing day

The United Nation (UN) announces that the Global Handwashing Day occurs on 15 October of each year. It aims to motivate people around the world to be aware of and to improve their handwashing habits by washing hands with soap regularly. This habit is a simple way to prevent infectious diseases. Every year, the Department of Health under the Ministry of Public Health organizes an event for the Global Handwashing Day campaign. The department suggests a procedure for handwashing as follows.

1. Rub palms together.
2. Rub the backs of both hands.
3. Interlace fingers and rub the hands together.
4. Interlock fingers and rub the back of fingers of both hands.
5. Rub thumb in rotating manner followed by the area between index finger and thumb.
6. Rub fingertips on palm for both hands.
7. Rub both wrists in a rotating fashion.

Each step has to be done five times alternatively between the left and the right hand.

Design an algorithm for washing hands concerning the above procedure.

Bibliography

David D. Riley and Kenny A. Hunt. (2014). Computational Thinking for the Modern Problem Solver. CRC Press Taylor & Francis Group. USA.

Karl Beecher. (2017). Computational Thinking. A Beginners Guide to Problem-Solving and Programming. BCS Learning & Development Ltd.

BBC Bitesize. (2018). Abstraction. Retrieved June 5, 2018, from https://www.bbc.com/education/guides/zp92mp3/revision/1

Code.org. (2018). Computational Thinking. Retrieved June 10, 2018, from https://code.org/curriculum/course3/1/Teacher

Google Computational Thinking for Educators. (2018). Computational Thinking. June 19, 2018, from https://computationalthinkingcourse.withgoogle.com/unit

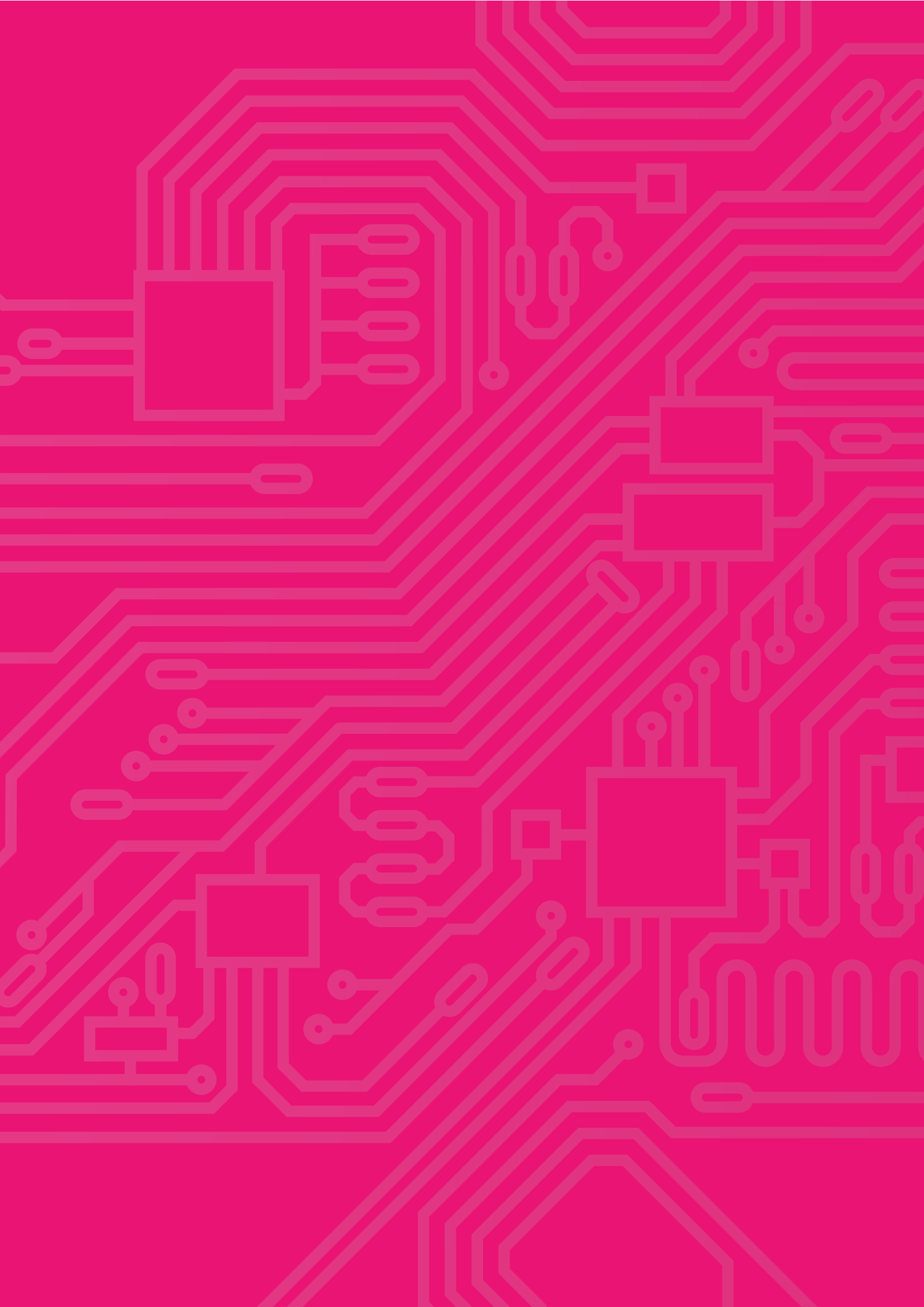
Chapter 2

Explore the KidBright World

Learning Objectives

At the end of the learning process of Chapter 2, students are able to:

1. Understand the meaning of computer programming or coding.
2. Gain basic knowledge of KidBright board.
3. Understand the components on a KidBright board.
4. Understand coding command series in a KidBright IDE program.
5. Understand the meaning of KidBright's command blocks.



Learning Contents

Meaning of computer programming

Computer programming or for short is the programming or coding process of designing, building and debugging a source code to create a set of commands for accomplishing a task on a computer. We can write source code in one or more computer languaes.

To be executable by the central processing unit of a computer, we requires a compiler or an interpreting program to convert the source code we have written into a machine language. For example, in Figure 2.1, the C compiler converts a source code of C language to a machine language.

! Computer programming is a blending of arts, sciences, mathematics, and engineering.

Figure 2.1 Complier converts source code in C language to a machine language.

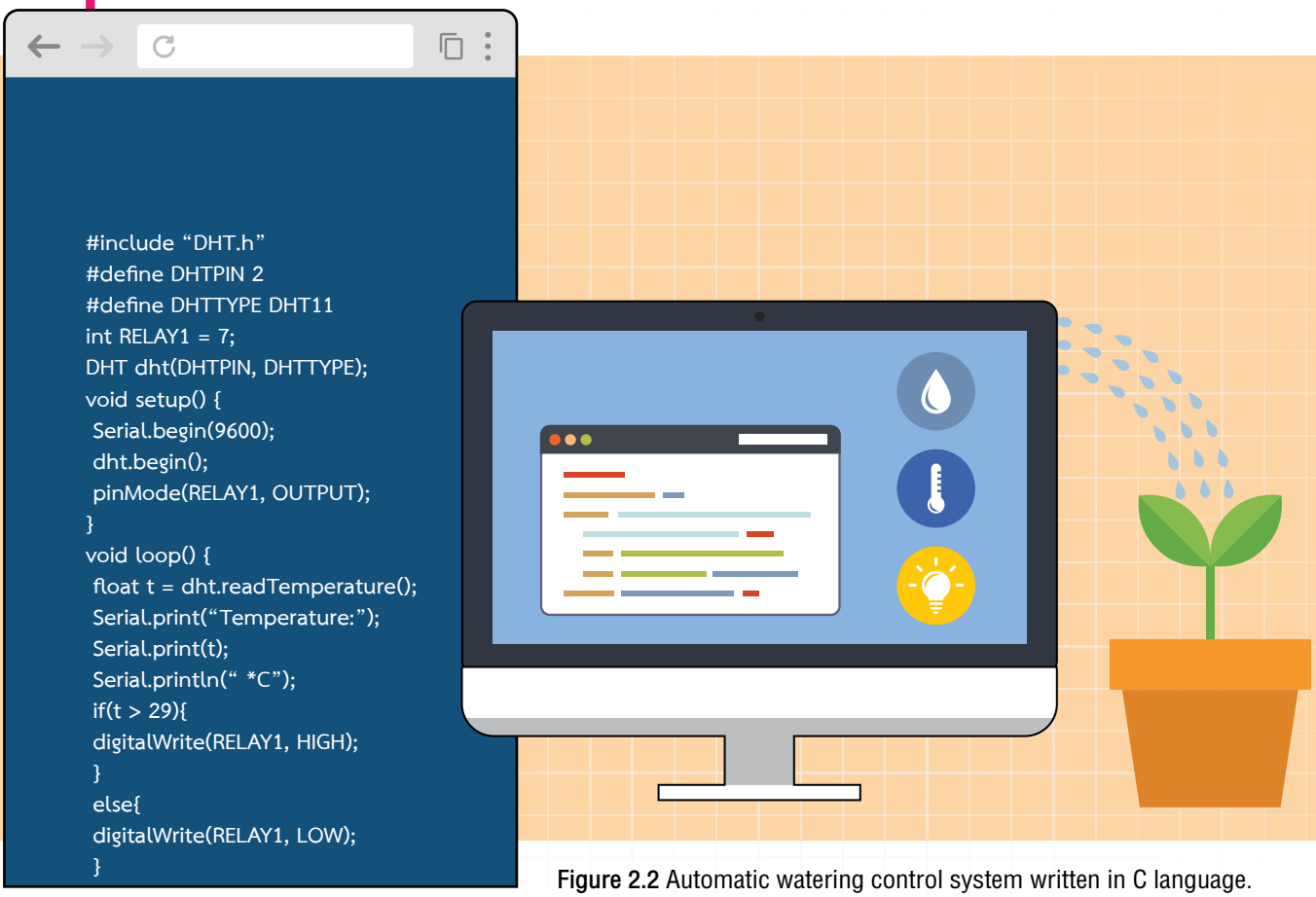


Figure 2.2 Automatic watering control system written in C language.

Creating command series in block format

A computer programmer has to memorize accurately several commands and the computer syntax, or ways of putting commands together so that a command series can be written correctly. These commands are words that can only be combined in fixed patterns. The written program cannot be executed (will not work) if there is a miswritten code. This is an obstacle for children to learn coding, especially in elementary and junior high schools since they have limited interest and are easily distracted.

To overcome this obstacle, block commands were created to make coding easy. Instead of memorizing commands and computer syntax, programmers need only to arrange block command symbols and put them in a correct sequential order. This helps the programmer to focus on a thinking procedure, rather than finding solutions for incorrect commands.

Figure 2.2 shows an example of coding in C language to read the measured values from a temperature sensor and then use these values to control watering of plants.

When the program in Figure 2.2 is written in the block language, it produces the same output (results), but this can be composed more easily with only a few blocks as shown in Figure 2.3. The block language is able to do the same work as C language.

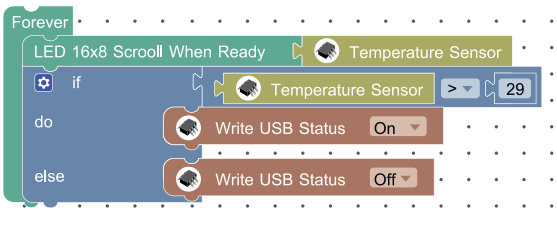
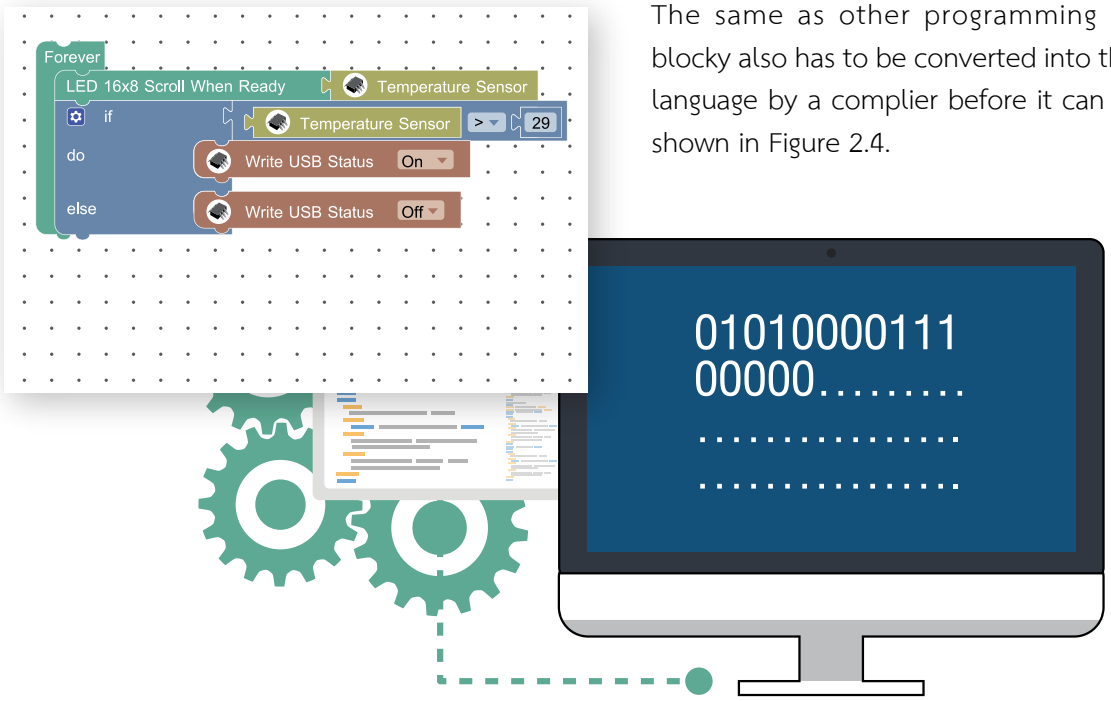


Figure 2.3 Automatic watering control system written in block language.

The block language or blocky is a new language developed by Google as a Visual Programming Language or Graphical Programming Language in which users arrange symbols on a screen to connect them correctly for processing in the sequential order.



The same as other programming languages, blocky also has to be converted into the machine language by a complier before it can be used as shown in Figure 2.4.

Figure 2.4 Complier converts source code of block language to the machine language.

Components of KidBright Board

A KidBright board is an embedded board used as mini-controlling equipment consisting of a microcontroller, a display monitor, a Real-time clock, a buzzer and simple sensors. The KidBright board executes coding commands written in block based programming language. The KidBright board is used as a tool for teaching computer programming. Learners can program it by using coding commands in the KidBright Integrated Development Environment program, or KidBright IDE in short. KidBright IDE can also be used on several operating systems, such as Windows, macOS, and Ubuntu. A set of written commands are sent to KidBright board to execute the program. It makes learners understand the real operation built in real time. For students to better understand KidBright board, definitions of the embedded board and the operating system are explained in the following sections.

What is an embedded board?

An embedded board is a small computer embedded in a device to make the device simpler through controlling software that is different from a data processing system in a normal computer. Embedded boards are widely used in electrical devices, vehicles, and communication equipment as shown in Figure 2.5. Since it is embedded in a device, it is called an embedded board. In general, the software to control the embedded board system can be developed with or without an operating system.

Figure 2.5 Embedded board.

Operating System (OS)

An Operating System or OS is a program that functions as a media connecting users, hardware and applications. Applications that run through OS effectively manage computer resources to accomplish various tasks, such as controlling and operating visual display units and other hardware units. OS allows applications and hardware to communicate through the sending and receiving of data as shown in Figure 2.6.

In general, operating systems are fundamental program for computers, mobile phones, laptops, PDA tablets, and other embedded systems. Operating systems control functions in these devices and connect users through applications. Examples of operating systems in computers are Windows, Linux, macOS and Solaris. Those in mobile phones are Windows Mobile, iOS, and Android, and those in embedded boards are QNX and FreeRTOS.

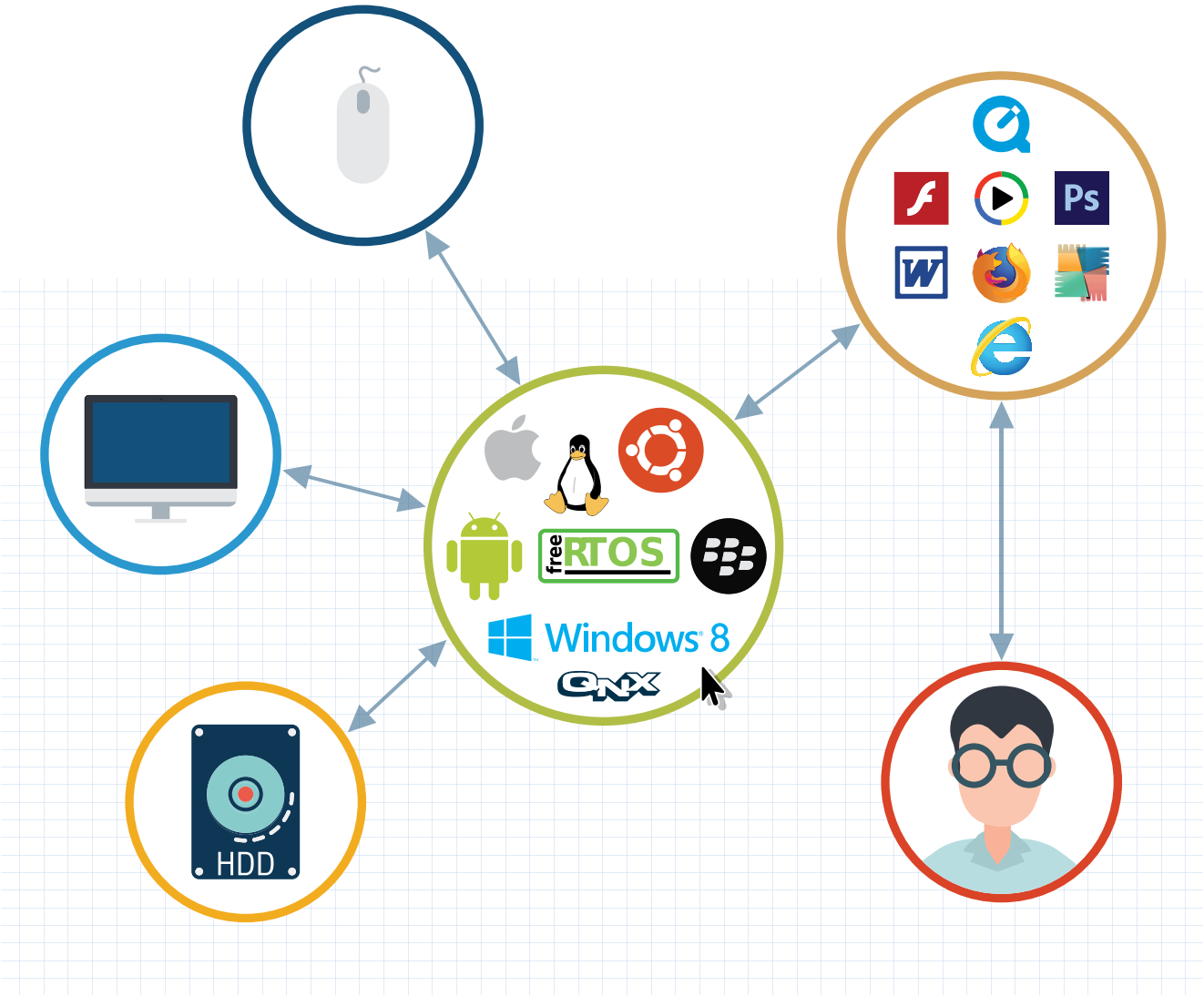
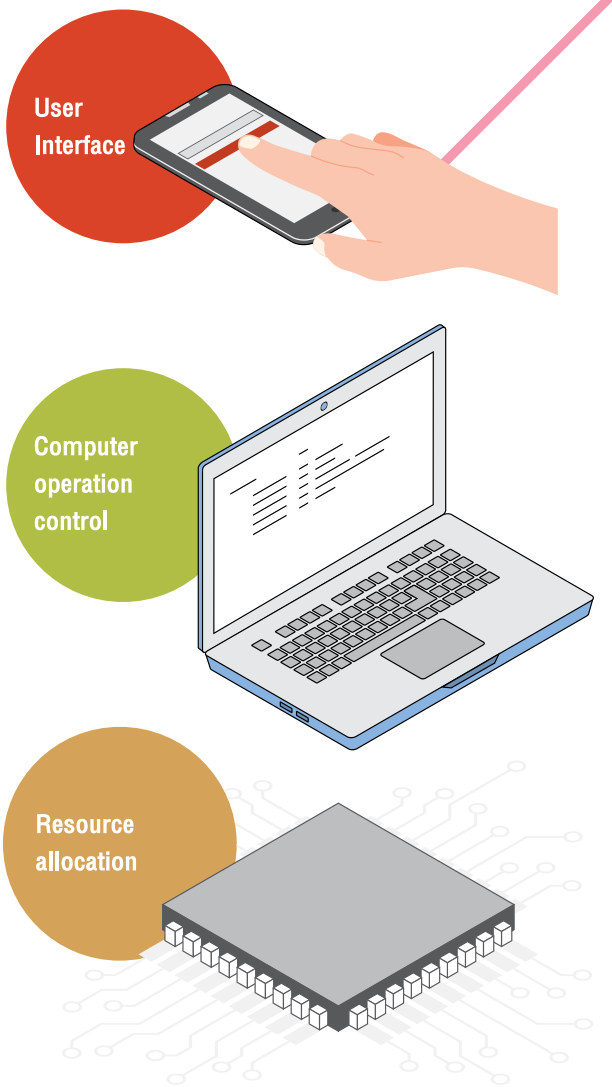


Figure 2.6 How an operating system works.

Functions of operating system



User Interface: One of the main objectives of creating an OS is to enable communication between users and hardware. Users purposefully give commands to hardware to accomplish certain tasks but they don't have to know how hardware is controlled and how it works in detail. The OS connects them and makes the communication easy. The interface is the means by which the user and a computer system interact, in particular via input devices and software.

Computer operation control: The function of OS is to control hardware operations instead of users through the Device Driver of each device.

Resource allocation: In order to work, a computer needs various resources such as a processing unit and a memory part. As resources are limited, they must be managed for maximum efficiency.

In general, the purpose of computer programming using different languages or block commands e.g. with KidBright is to create an application software. The created application software connects with the OS. To use or link the application to hardware, it is essential for the application software to be converted into a form of machine language which the OS can understand.

How KidBright works

KidBright consists of two parts namely an embedded KidBright and the KidBright IDE program. Learners can create command series through KidBright IDE by dragging and pasting the required command block symbols on the screen. KidBright IDE compiles the program in block language into machine language. The compiled program is then sent to the KidBright board to execute the program for the desired application, e.g. controlling watering of plants by detecting humidity level, or switching a light on or off at specified times.

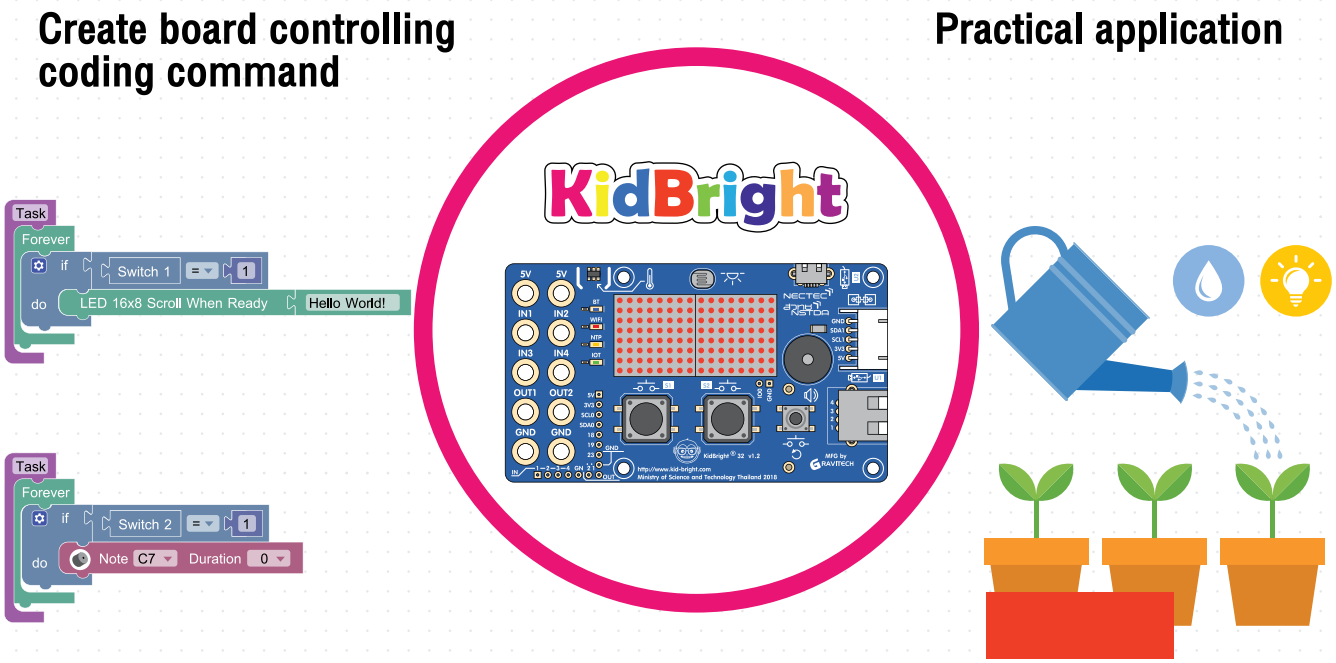


Figure 2.7 Overview of KidBright.

!

Compile is to convert a computer program in any programming languages to a machine language or to a command code that is applicable with the functioning control part of the embedded board.

KidBright embedded board

The KidBright board used in "Coding at School" Thailand project is the second generation of the board and is called "KidBright32". KidBright32 uses a small controlling equipment called microcontroller (No. ESP32) to control the board. KidBright32 has a display monitor, a Real-time clock, a buzzer and some sensors. The board receives the command series from a KidBright IDE through a USB cable. The KidBright board can connect with additional external sensors through IN1-IN4 signal ports, as shown in Figure 2.8-2.9.

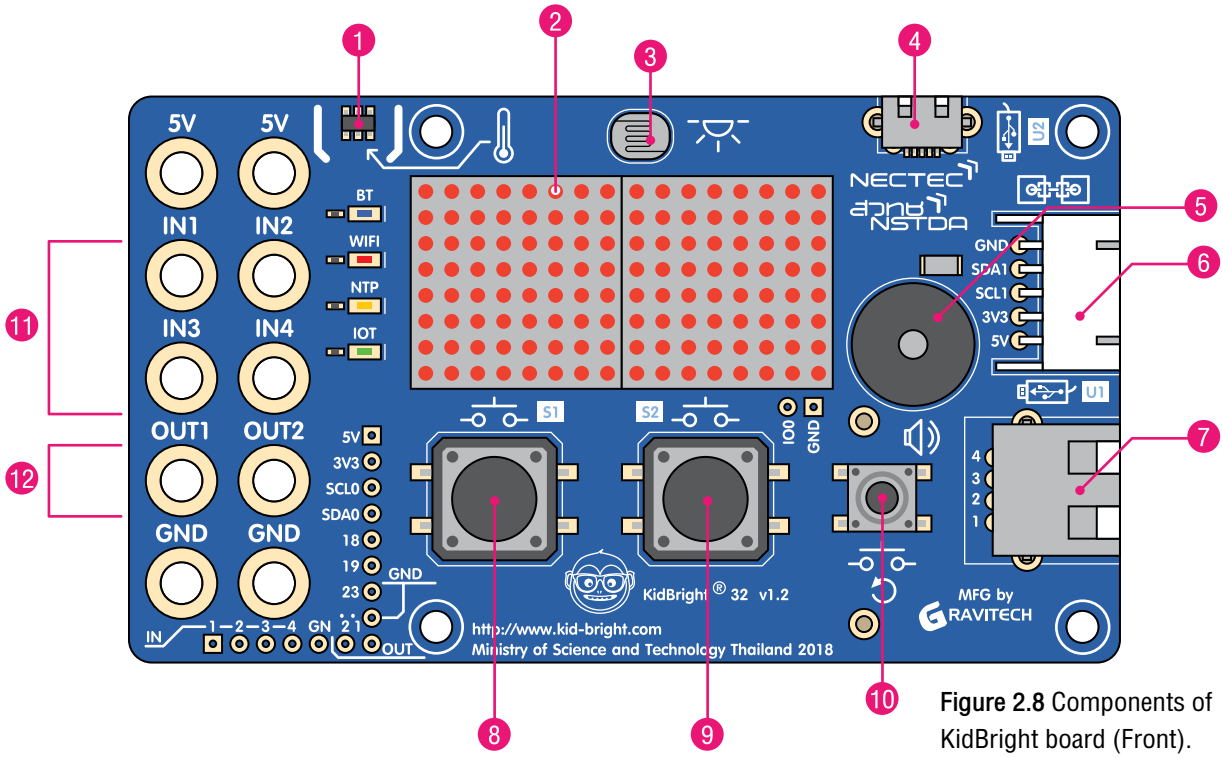
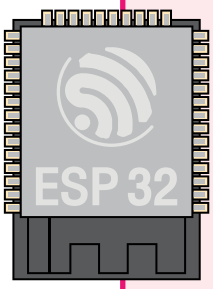


Figure 2.8 Components of KidBright board (Front).

1. Temperature senser
2. LED display
3. Light sensor
4. Micro USB port
5. Buzzer
6. Connector
7. USB port
8. Switch 1
9. Switch 2
10. Reset switch
11. Input signal 1-4 ports
12. Output signal 1-2 ports

!

The "Coding at School" project is run by the National Electronics and Computer Center, NSTDA with the goal of distributing 200,000 KidBright board to schools across Thailand. The project formally launched in May 2018, funding by the Thai government through The Ministry of Science and Technology.



KidBright32

KidBright32, a combination of KidBright board and IoT (Internet of Things), helps sending data through a network system for controlling equipment in long distances remotely such as watering plants using a mobile phone.

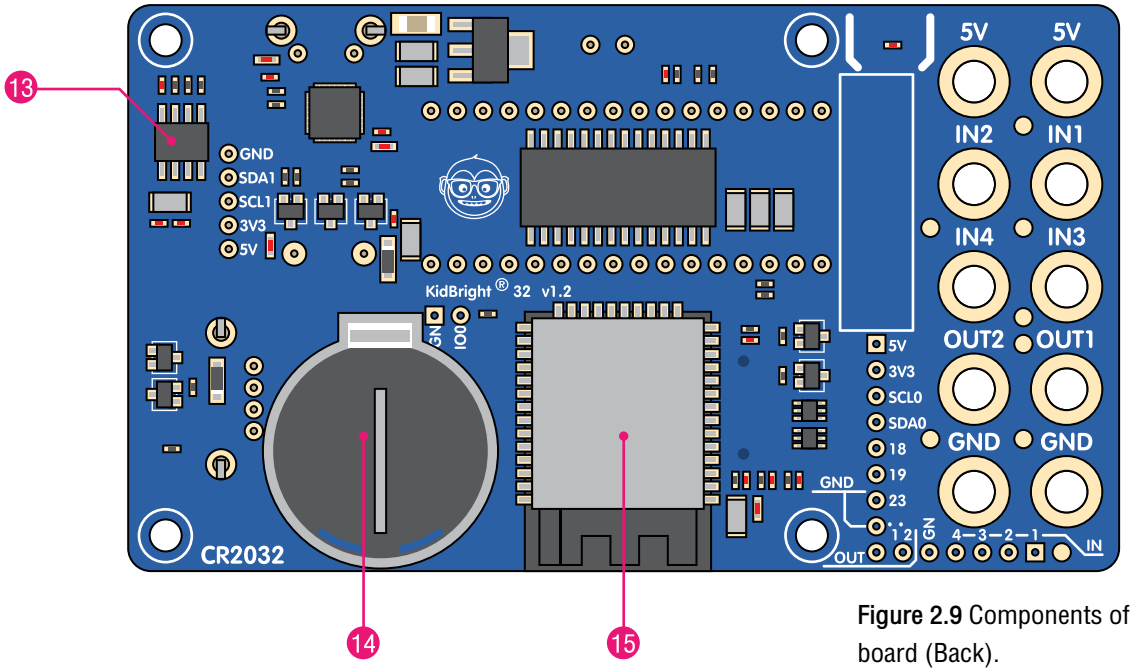


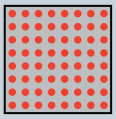
Figure 2.9 Components of KidBright board (Back).

13 Real-time clock 14 Battery compartment 15 Controlling part

Functions of various components on a KidBright Board

1 

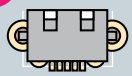
Temperature sensor
Measures temperature around KidBright board.

2 

LED display
Displays text and pictures.

3 

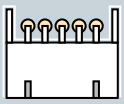
Light sensor
Measures light around KidBright board.

4 

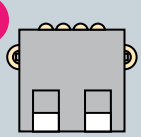
Micro USB port
Receives electric current from the generator and data from computer.

5 

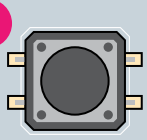
Buzzer
KidBright sound generator.

6 

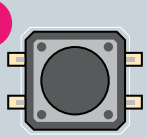
Connector
Connects KidBright to supporting board.

7 

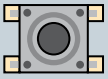
USB port
Connects to USB electrical device, e.g. fan.

8 

Switch 1
Connects KidBright board, e.g. monitor displays data when pressed.

9 

Switch 2
Controls KidBright board, e.g. plays music when pressed.

10 

Reset switch
Resets and restarts KidBright board.

11 

Input signal 1-4 ports
Receives digital input signal from connected external device.

12 

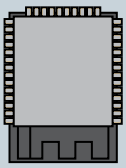
Output signal 1-2 ports
Sends digital output signal to connected external device.

13 

Real-time clock
Battery-operated Real-time clock.

14 

Battery compartment
Houses battery for Real-time clock.

15 

KidBright controlling part
Controls KidBright board by Microcontroller ESP32 for connecting to internet via Wifi and Bluethooth.

Coding command program

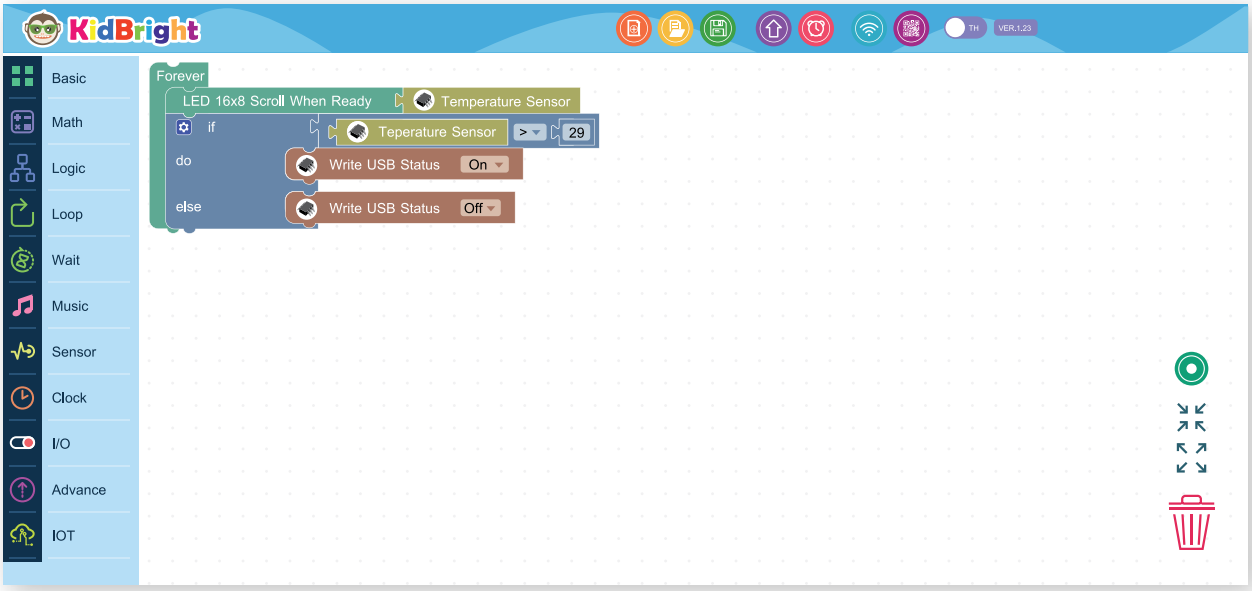


Figure 2.10 KidBright IDE main windows.

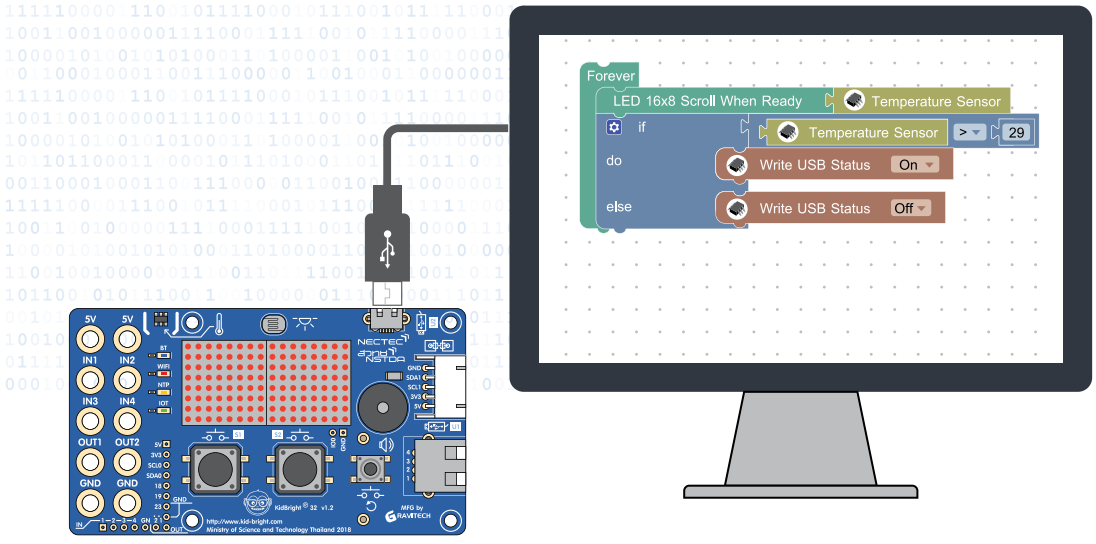
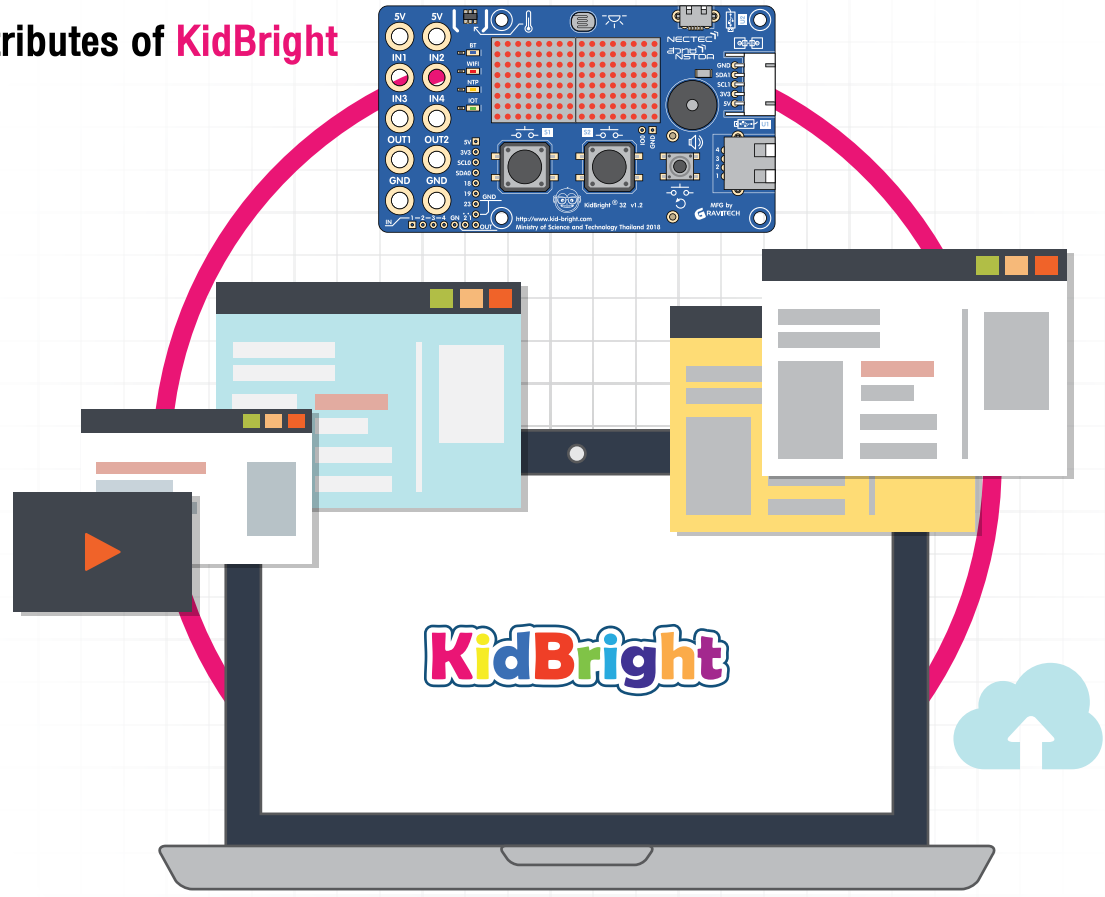


Figure 2.11 Coding command program functioning via USB cable.

As illustrated in Figure 2.10–2.11, KidBright IDE is used to create a command series using block based programming. Users drag and order command blocks to make a command series. After the command series is complete, KidBright IDE will compile the command series to a command code that is communicated to the board through a USB cable.

Attributes of KidBright



Supports computer applications via KidBright IDE coding command series.

Supports event-driven programming.

Supports multitasking.

Supports various sensor connections.

Supports connections between devices using Internet of Things.

Event-driven Programming is a method of computer programing where a sequence of programing is determined by events, for example, working when a user presses a button.

Multitasking Programming is a method of computer programming for parallel working. It allows the processing of multiple tasks at the same time without waiting for any one task to finish.

Working procedure

Necessary equipments

- A personal computer using Windows or macOS.
- A KidBright embedded board.
- A micro USB cable.

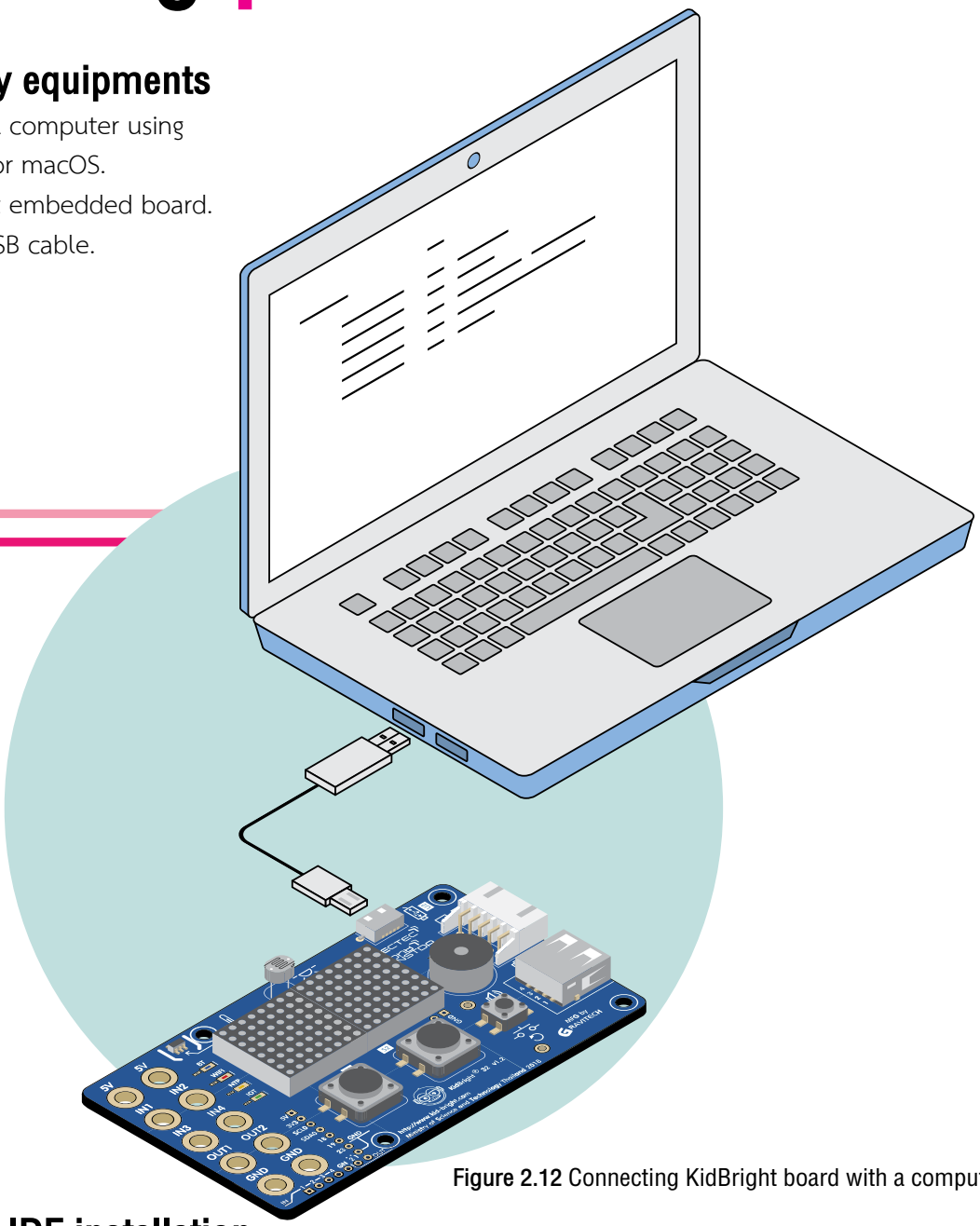


Figure 2.12 Connecting KidBright board with a computer.

KidBright IDE installation

1. Download KidBright IDE from www.kid-bright.org
2. Install KidBright IDE
   - Windows: Double click KidBright IDE-setup.exe file.
   - macOS: Extract KidBright IDE.app.zip file, copy KidBrightIDE.app and paste it into the application folder.

Start to work

Step 1: Connect a KidBright board to a computer as demonstrated in Figure 2.12 by using a micro USB cable. Connect the large connector end to USB port of the computer and connect the other end to the micro USB port of the KidBright board.

Step 2: Start the KidBright IDE command program.

For Windows operating system

Double click the KidBright IDE icon displayed on the computer desktop as shown in Figure 2.13 and KidBright IDE will appear as shown in Figure 2.15.

For macOS operating system

Double click the KidBright IDE icon displayed on Applications as shown in Figure 2.14 and KidBright IDE will appear as shown in Figure 2.15.

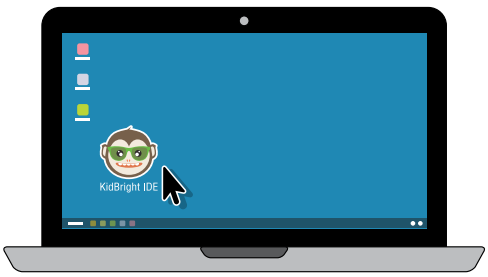


Figure 2.13 KidBright IDE icon on the desktop screen.

Figure 2.14 KidBright IDE icon on the Applications screen.

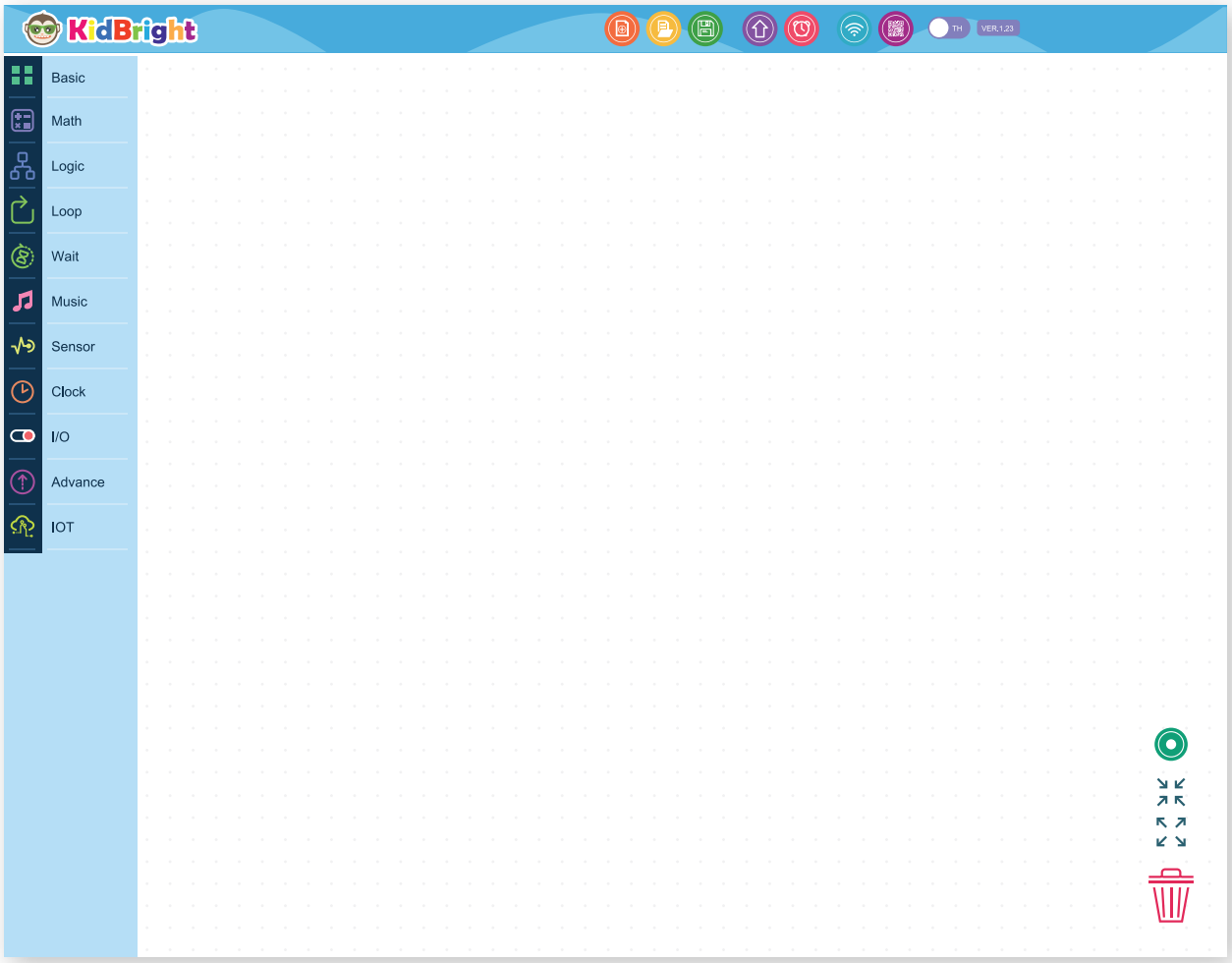


Figure 2.15 KidBright IDE coding command program.

Components of KidBright IDE

The main window of KidBright IDE as shown in Figure 2.16 comprises the following:

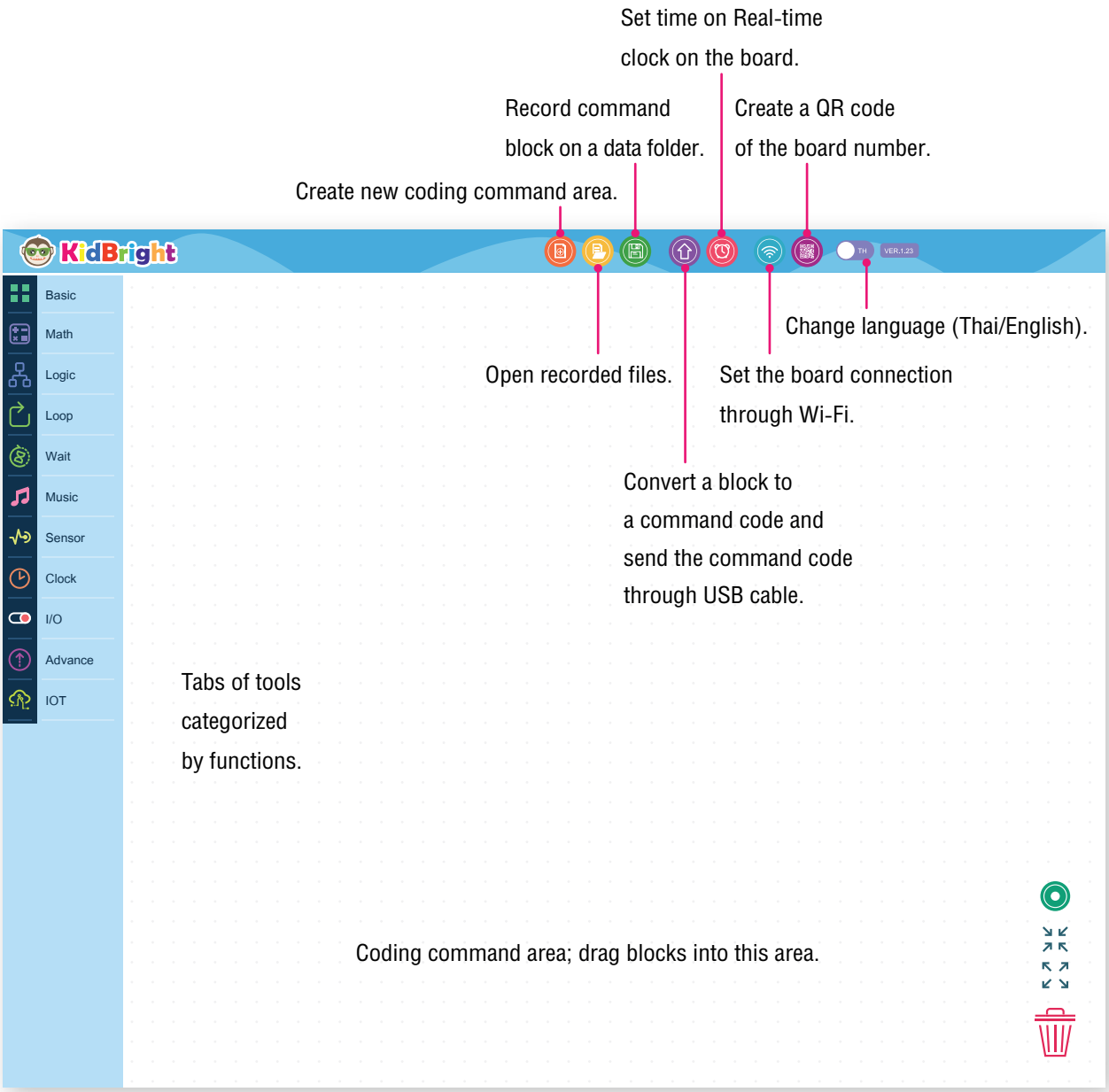


Figure 2.16 Main window of KidBright IDE coding command program.

KidBright board controlling command in blocks

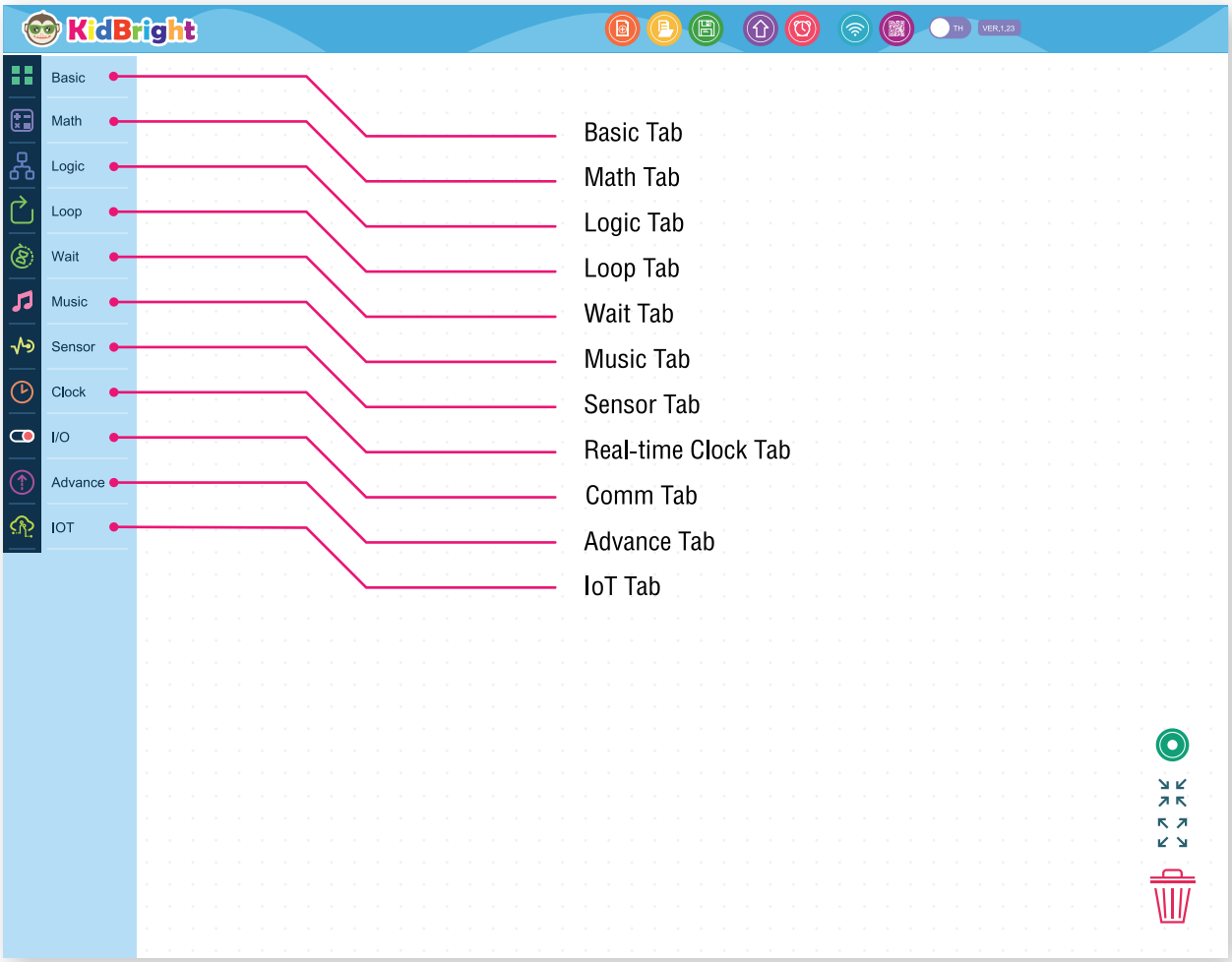


Figure 2.17 Tabs of tools categorized by functions.

Basic Tab

The Basic Tab contains several command blocks that are frequently used, such as displaying texts, delaying time, and looping as shown in Figure 2.18.

As shown in Figure 2.18, eight command blocks on the Basic tab function as follows:

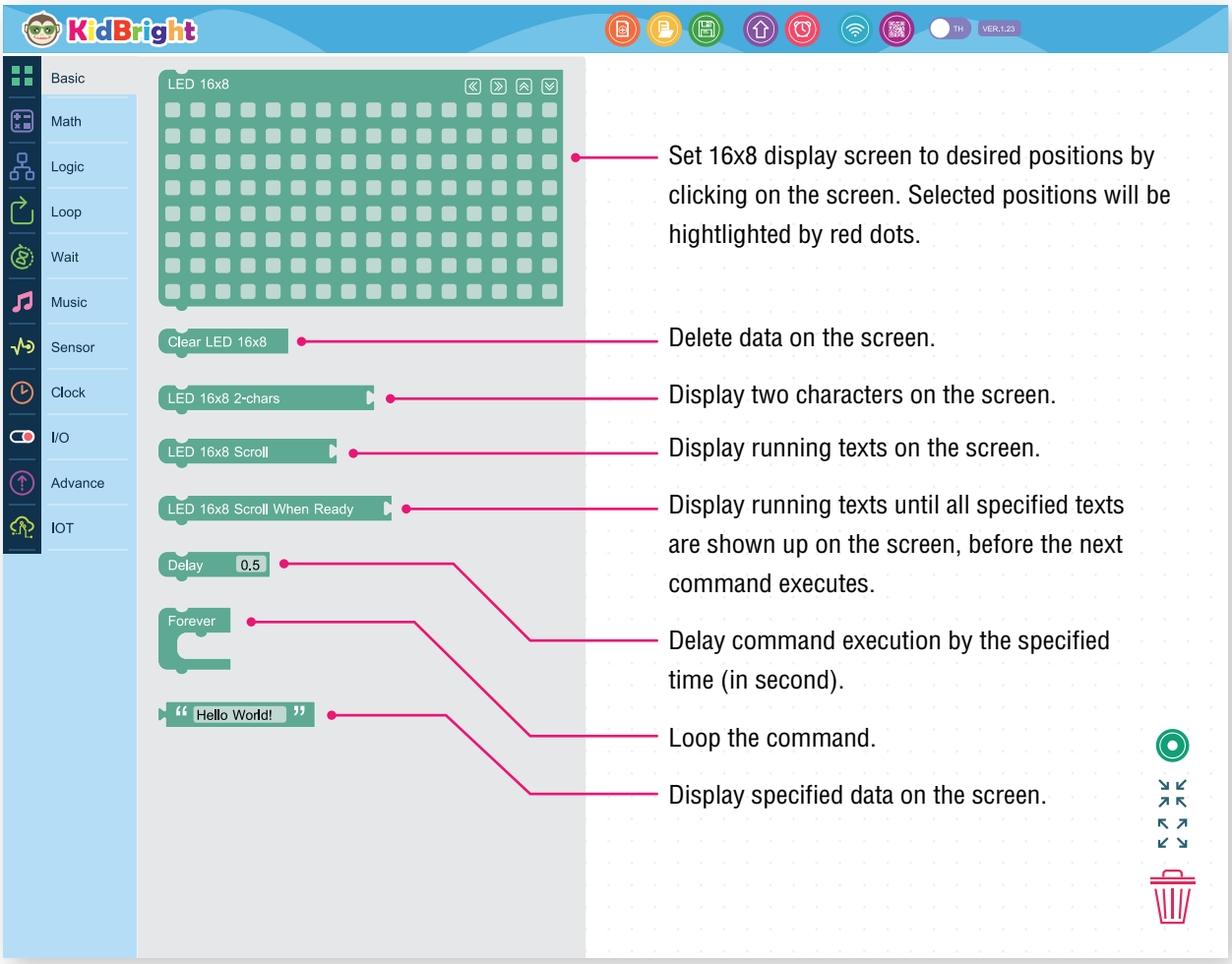


Figure 2.18 Command blocks on a Basic tab.

Math Tab

Math tab contains command blocks used to set constant variable, specify a variable, or conduct mathematical calculation as shown in Figure 2.19.

As shown in Figure 2.19, four command blocks on the Math tab function as follows:

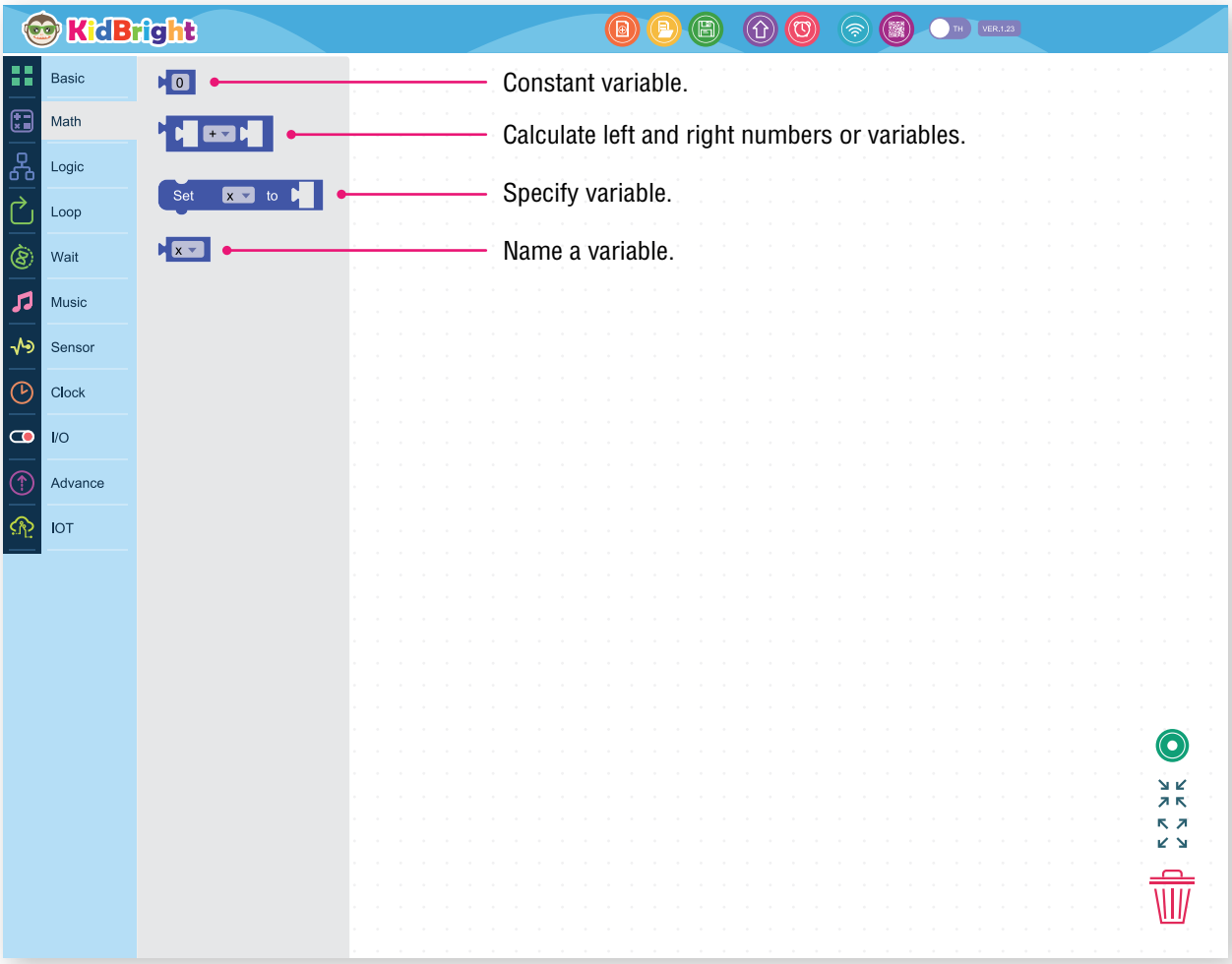


Figure 2.19 Command blocks on Math Tab.

Logic Tab

The Logic tab contains command blocks defining conditions and checking the status of Switch 1 and Switch 2 as demonstrated in Figure 2.20.

As shown in Figure 2.20, eleven command blocks on the Logic tab function as follows:

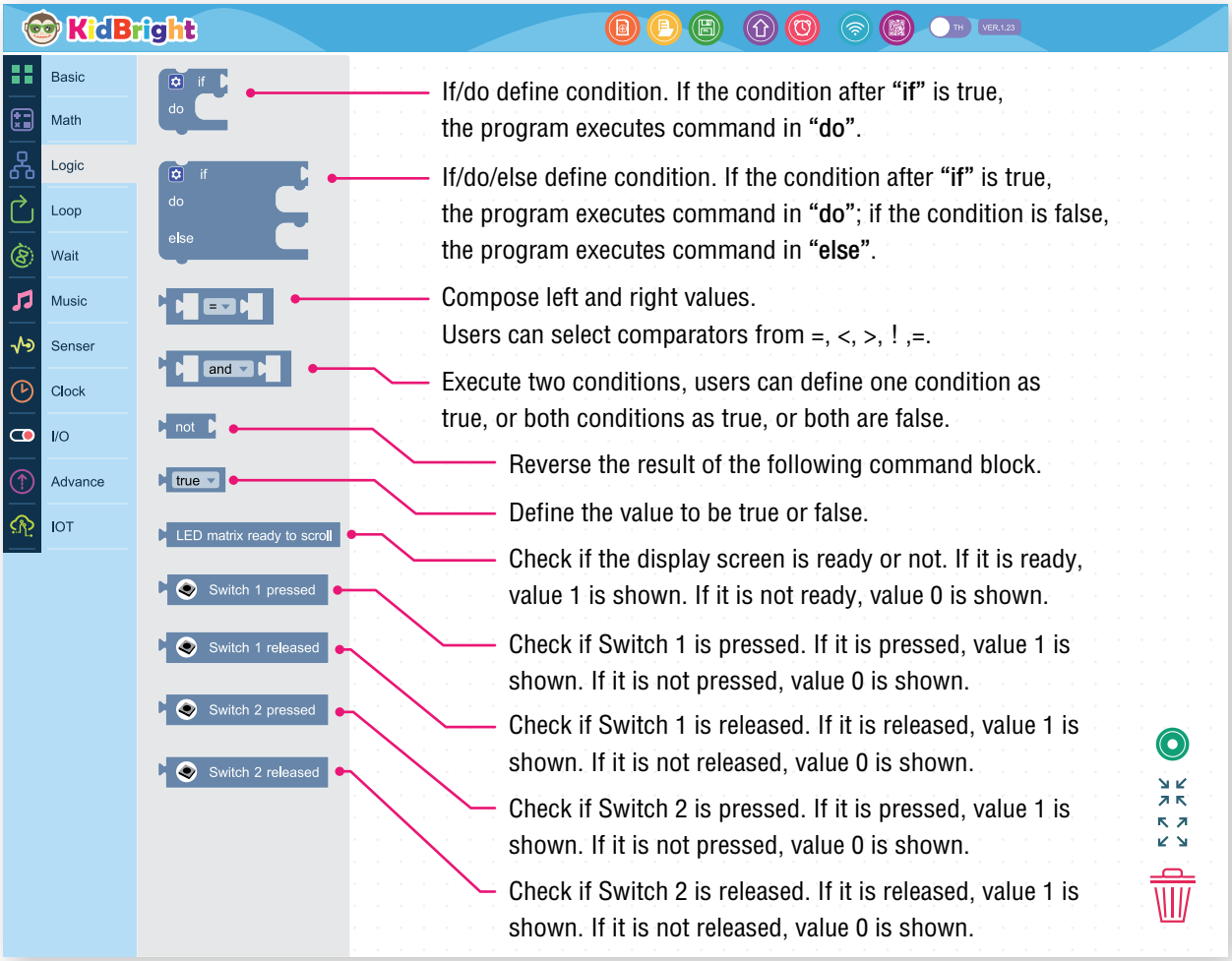


Figure 2.20 Command blocks on Logic tab.

Loop Tab

The Loop tab contains command blocks repeating the command in a cycle or loop as shown in Figure 2.21.

As shown in Figure 2.21, three command blocks on the Loop tab function as follows:

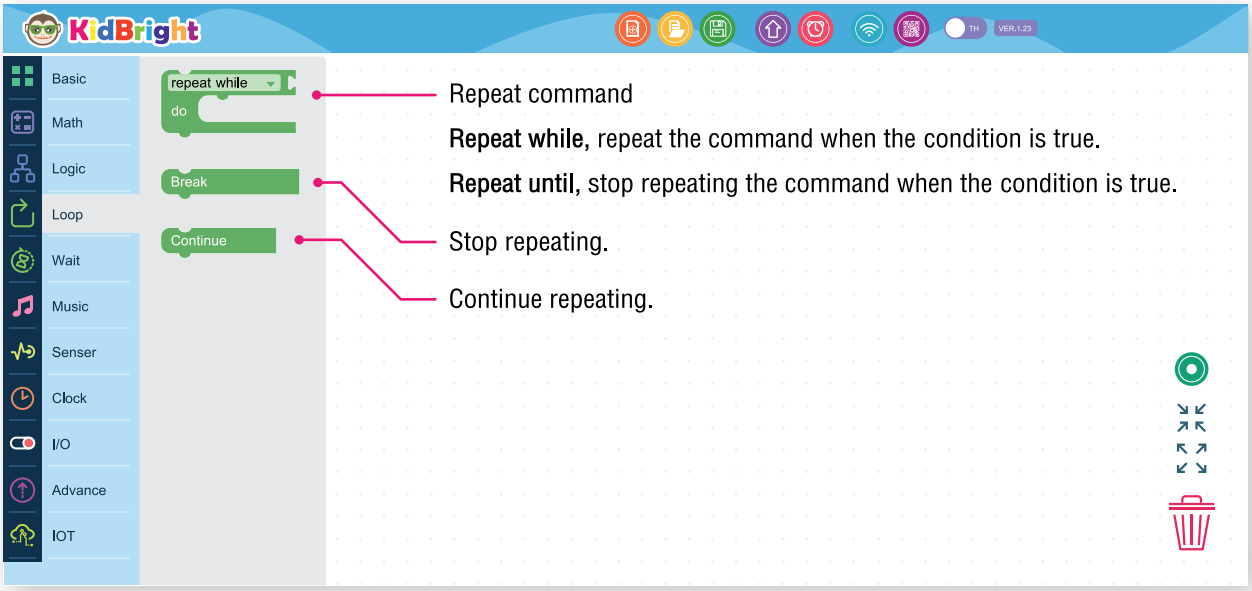


Figure 2.21 Command blocks on Loop tab.

Wait Tab

The Wait tab contains command blocks to temporarily stop working and wait until a new situation arises, such as stop and wait until Switch 1 is pressed as shown in Figure 2.22.

As shown in Figure 2.22, five command blocks on the Wait tab function as follows:

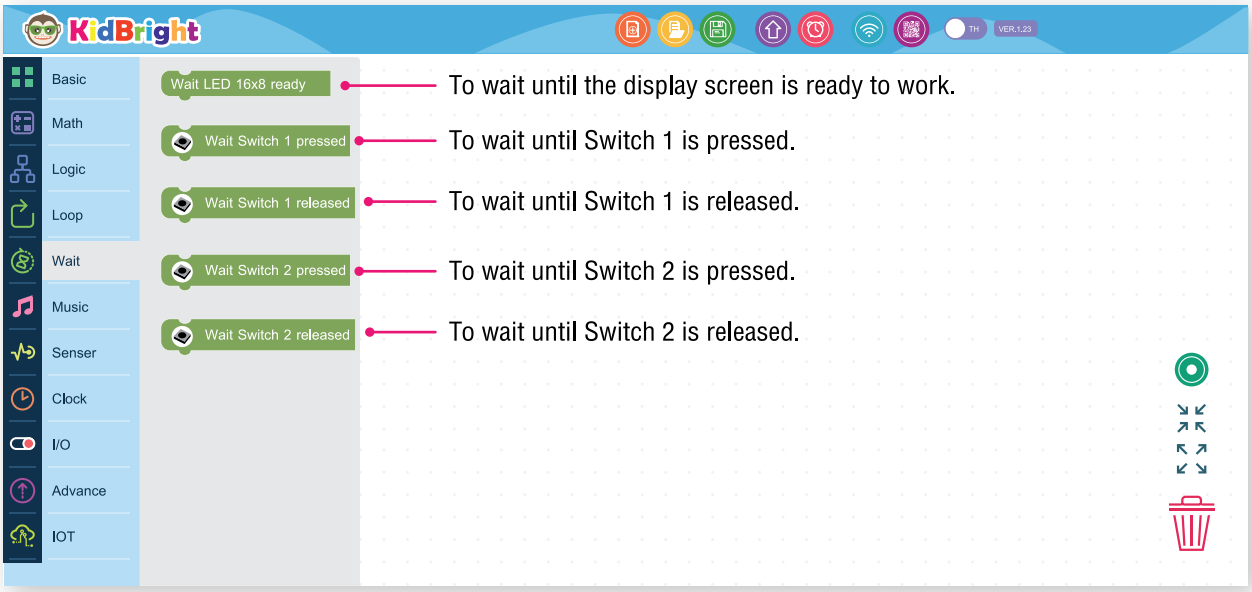


Figure 2.22 Command blocks on Wait tab.

Music Tab

The Music tab contains command blocks that control the buzzer to produce sounds as specified musical notes, as shown in Figure 2.23.

As shown in Figure 2.23, five command blocks on Music Tab function as follows:

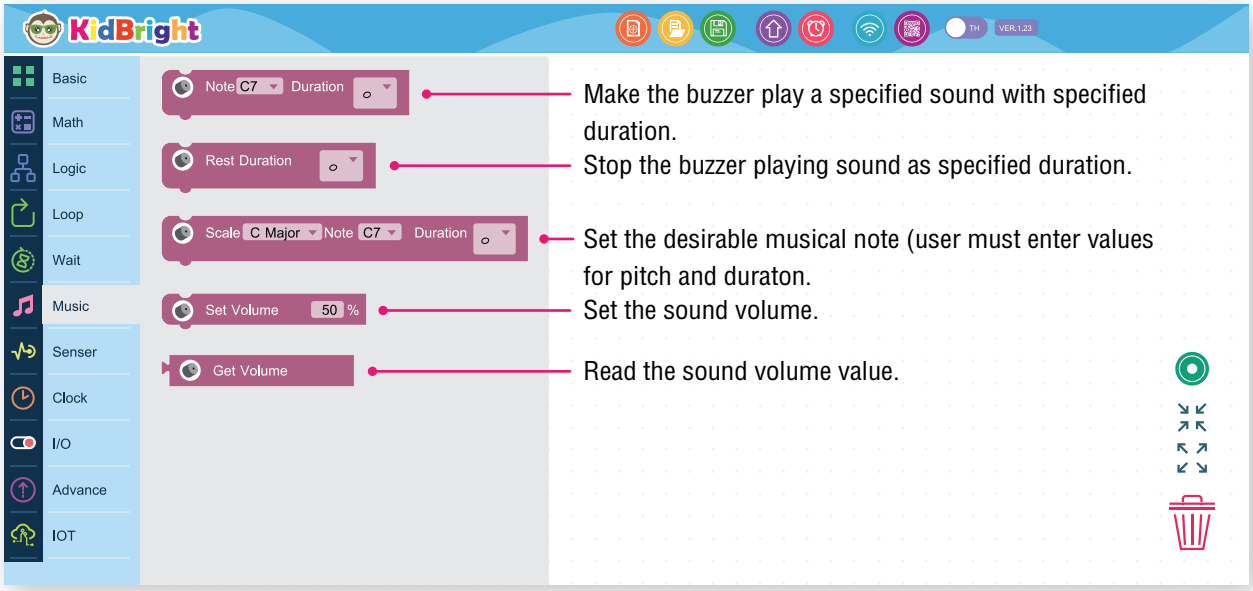


Figure 2.23 Command blocks on the Music tab.

Sensor Tab

The Sensor tab contains command blocks for using light or temperature sensors and checking Switch status as shown in Figure 2.24.

As shown in Figure 2.24, four command blocks on the Music tab function as follows:

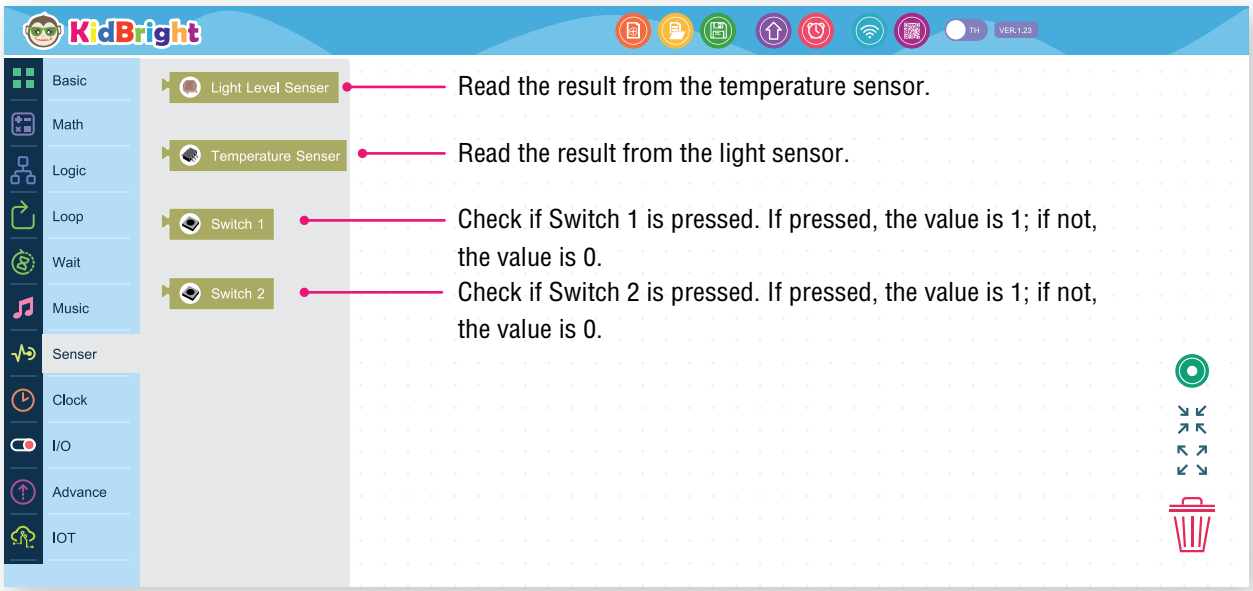


Figure 2.24 Command blocks on the Sensor tab.

Real-time Clock Tab

The Real-time clock tab contains command blocks involving with the Real-time clock, such as day, month, year and time as shown in Figure 2.25.

As shown in Figure 2.25, nine command blocks on the Real-time clock tab function as follows:

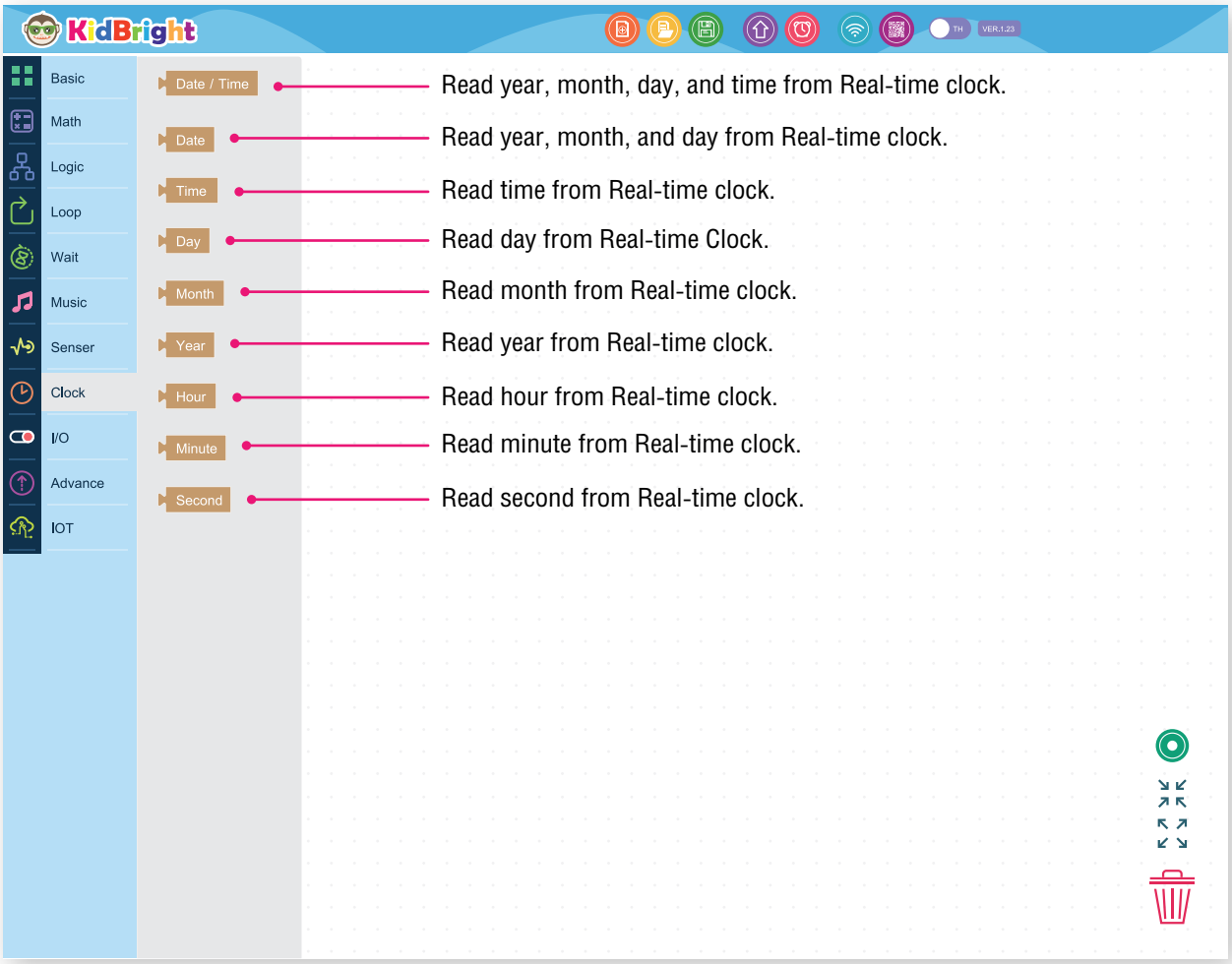


Figure 2.25 Command blocks on the Real-time clock tab.

Comm Tab (I/O)

The Comm tab contains command blocks for reading and controlling USB status as shown in Figure 2.26.

As shown in Figure 2.26, seven command blocks on the Comm tab have their function as follows:

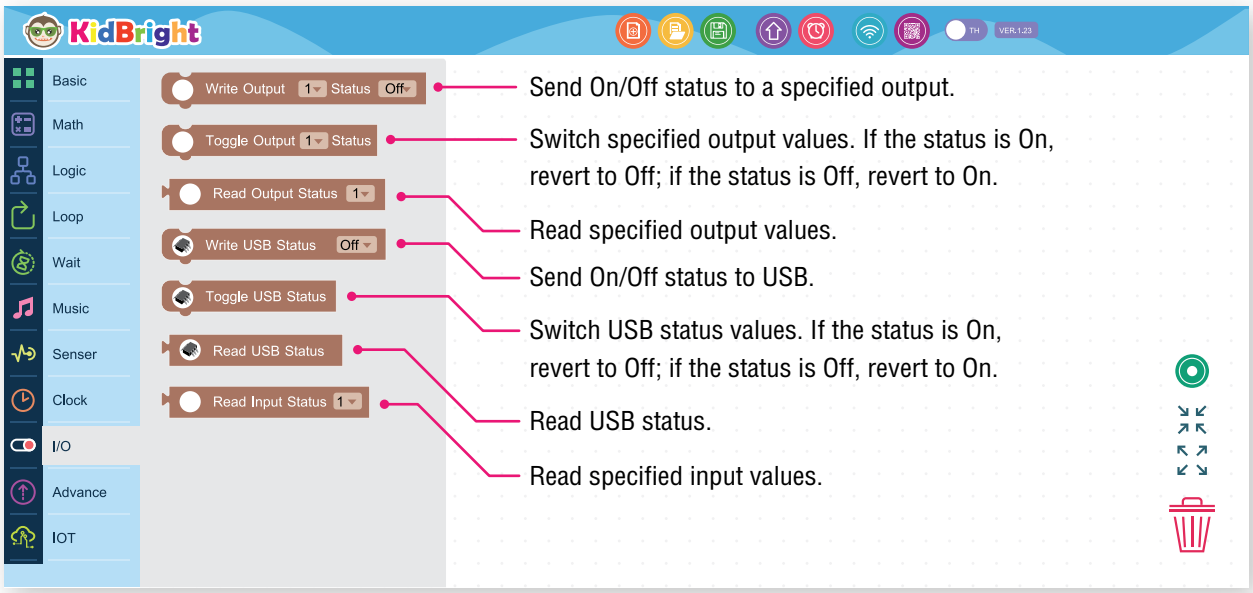


Figure 2.26 Command blocks on the Comm tab.

Advance Tab

The Advance tab contains command block working on multitasking function as shown in Figure 2.27.

As shown in Figure 2.27, a command block on the Advance tab function as follows:

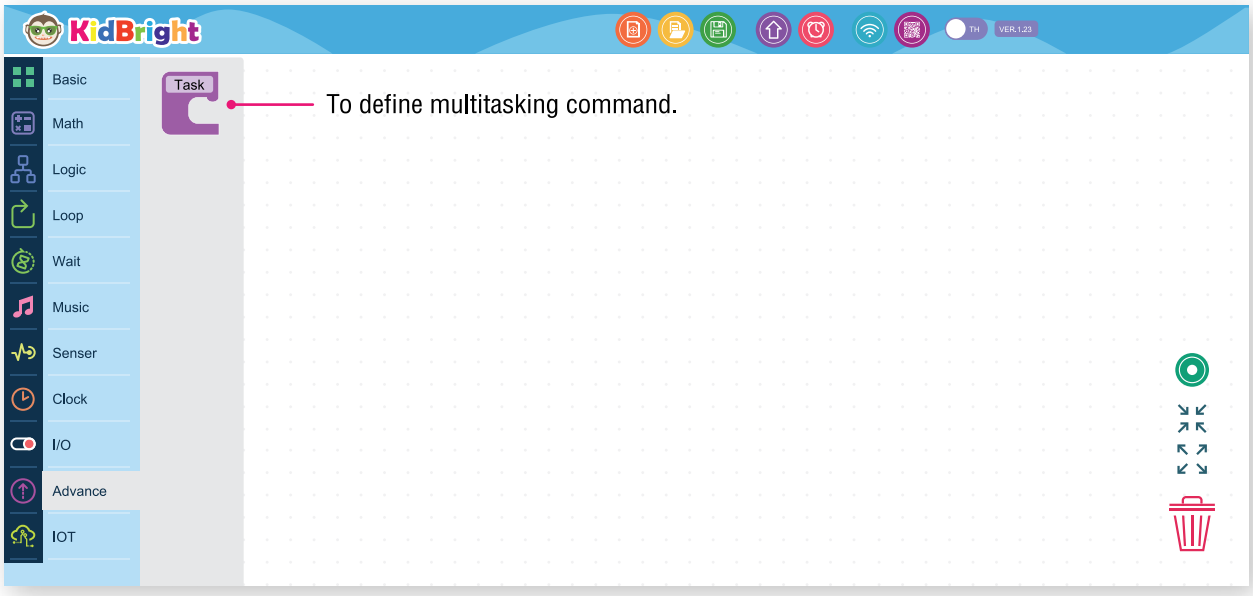


Figure 2.27 Command block on the Advance tab.

Computer programming testing

To code a command to display “1” on the screen of the KidBright board.

Step 1

1. Start from the KidBright IDE and navigate to the command tabs on the left hand side.
2. Drag ‘LED 16x8’ block from Basic tab (Block in the yellow frame), as shown in Figure 2.28.
3. Paste this command block on the programming area, as shown in Figure 2.29. Each small dot in the ‘LED 16x8’ block represents a corresponding dot on the KidBright board’s display screen.

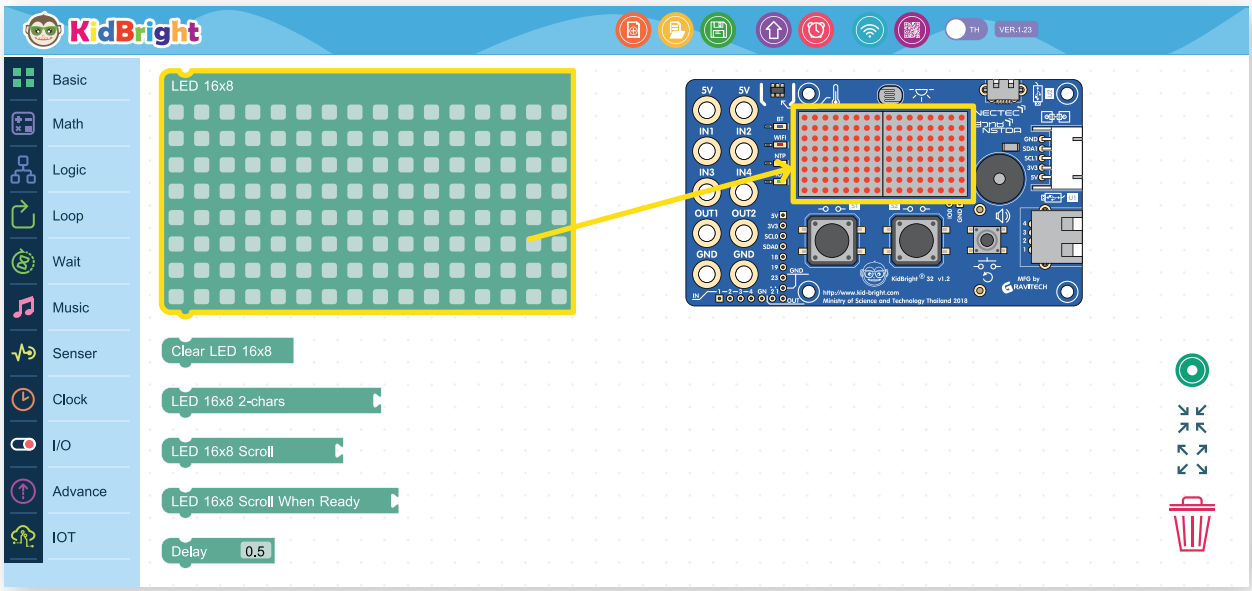


Figure 2.28 Selecting ‘LED 16x8’ command block from the Basic tab.

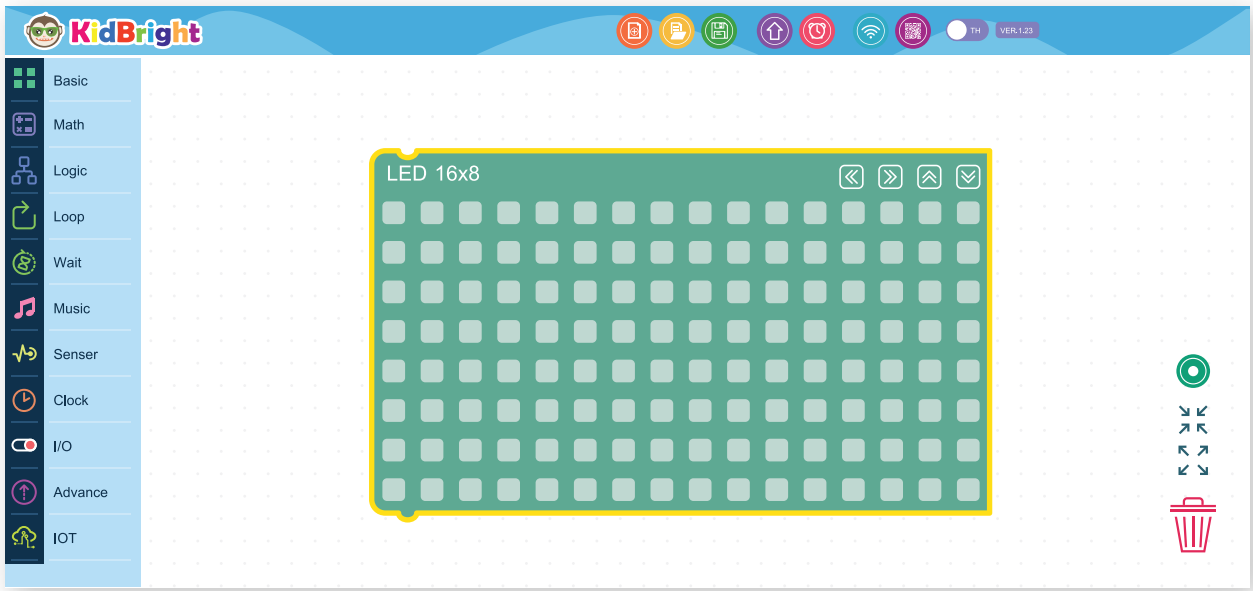


Figure 2.29 Pasting ‘LED 16x8’ command block onto programming area.

Step 2 Select small dots on the 'LED 16x8' command block by clicking on them. Once selected, the dots turn red. Select dots to form "1" as shown in Figure 2.30. Don't worry if you click a wrong dot. You can deselect the dot by clicking on it again. The selected red dots on the 'LED 16x8' block correspond to positions of dots on the KidBright board display screen.

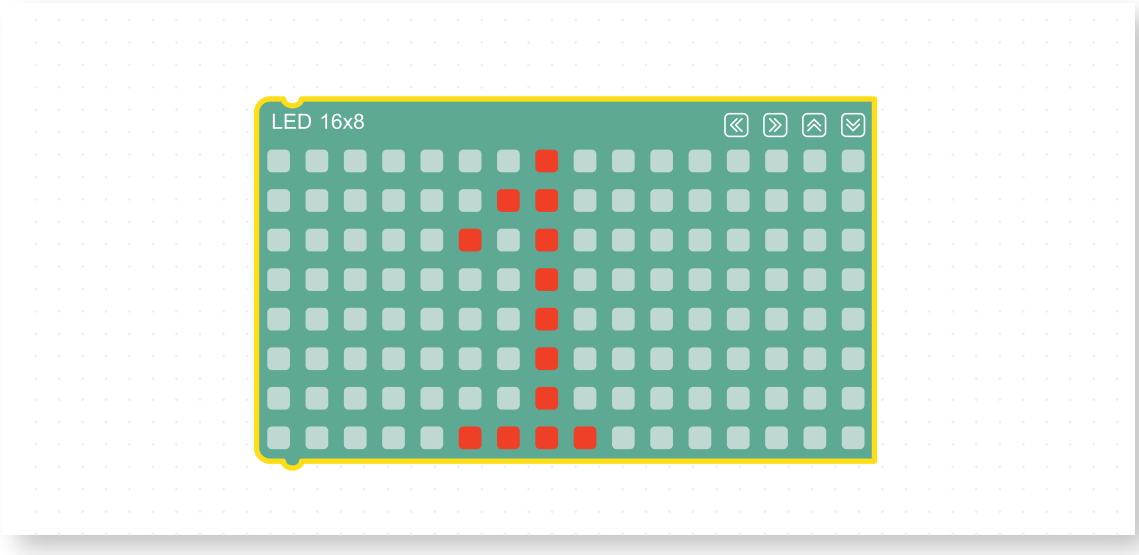


Figure 2.30 Selecting red dot positions to shown "1".

Step 3 Click on the **Program Build** button, as shown in Figure 2.31.

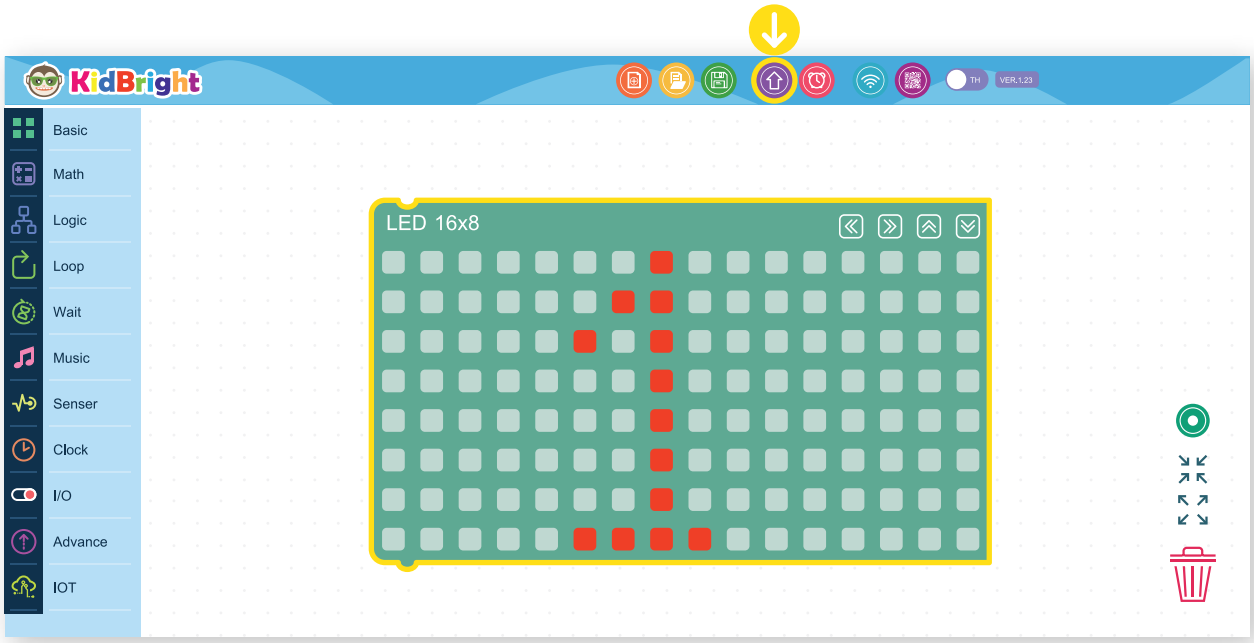


Figure 2.31 The Program Build button.

Step 4 If the code was entered correctly, the command series will be compiled into machine code automatically after the Program Build button is clicked. Once compiling is complete, the machine code will be sent to the KidBright board via a USB cable and "1" will appear on the KidBright board display screen.

If the code was incorrect, an error message will appear. All errors need to be corrected. Note that user can click exit button on the top-right to close the program.

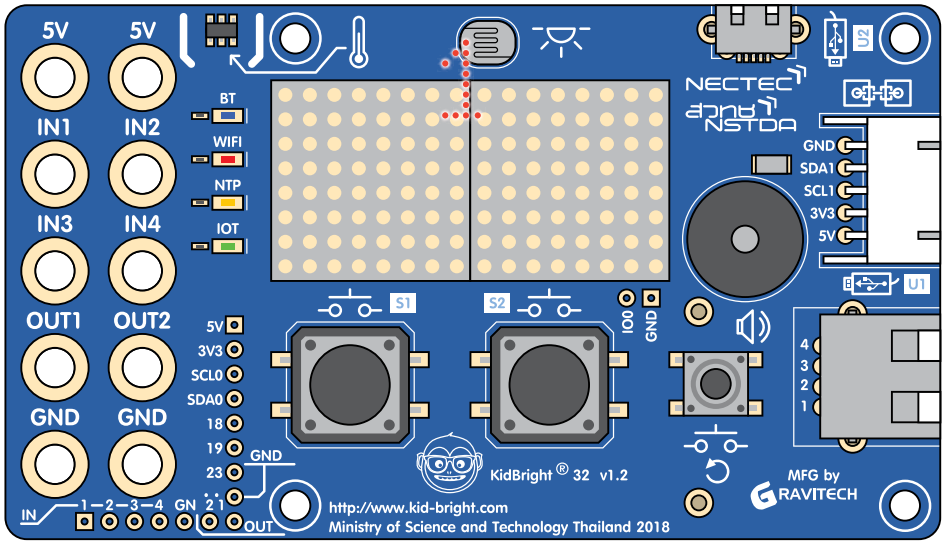
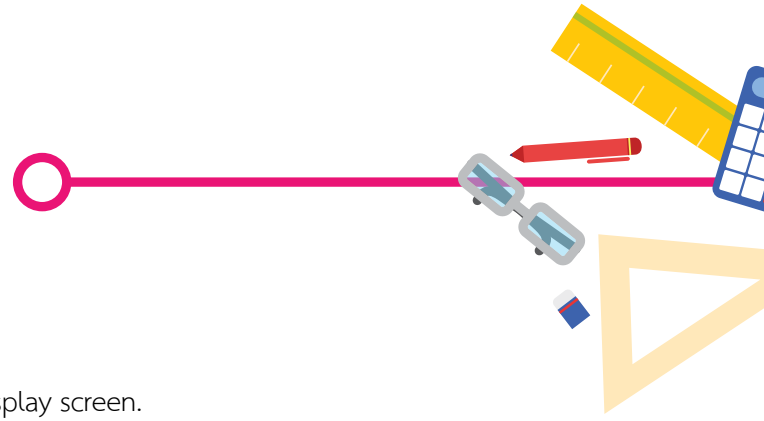


Figure 2.32 Successful program execution; "1" appears on the KidBright board display screen.

Summary

This chapter covers the contents of computer programming or coding, block coding, details of KidBright board components, and the command tabs of KidBright IDE.

Activity

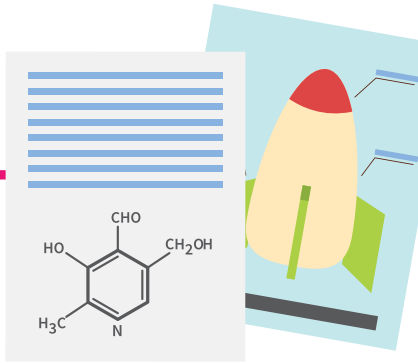


Activity 2.1

Write and execute a program for showing “10” on the display screen.

When you arrive at your favorite restaurant and find that it is closed, what would you do?

Exercise



Please explain the meaning of the following terms:

Embedded System

Operating System

Please explain the function of:

An embedded KidBright board

The KidBright IDE program

A light sensor on KidBright board

A temperature sensor on a KidBright board

Input signal ports on a KidBright board

Output signal ports on a KidBright board

Write a program for showing an image of smiling face on the display screen.

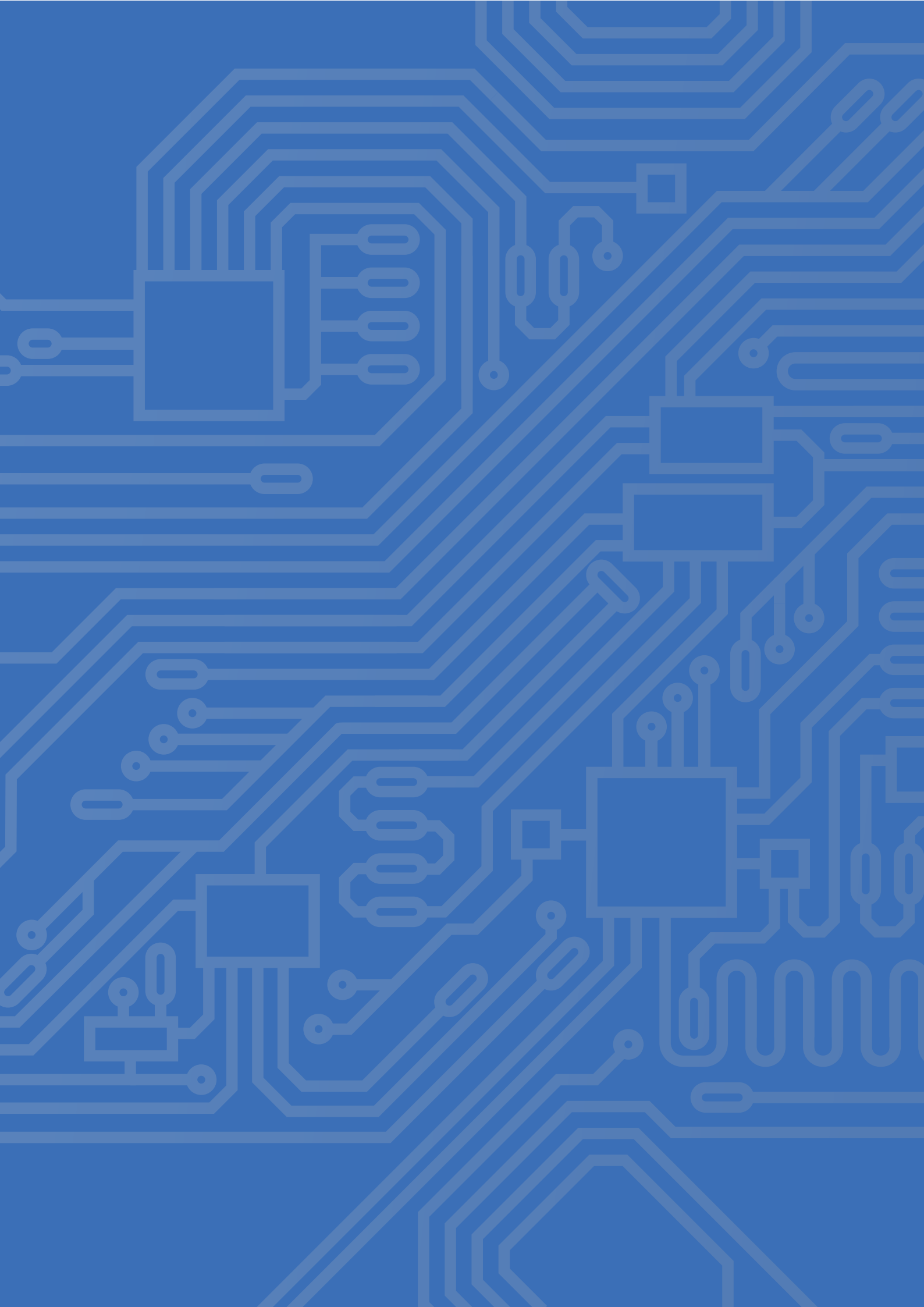
Chapter 3

Creating Moving Images with KidBright

Learning Objectives

Students are able to:

1. Write programs in KidBright IDE to work with the KidBright board.
2. Use the KidBright IDE to compile programs and operate on the KidBright board.
3. Create images using block commands.
4. Create images and moving images.



Learning Content

Digital image

A digital image consists of many dot pixels. Figure 3.1 shows an 8x8 dot-matrix display.

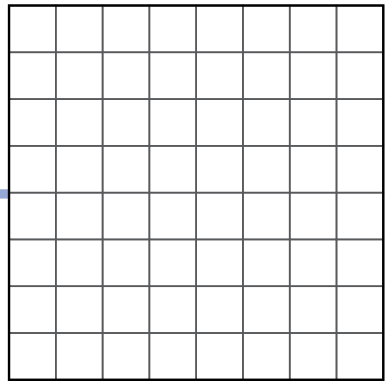


Figure 3.1 Dot-matrix display formed by many dot pixels.

An image is created by assigning colors to pixels. For example, on an 8x8 LED dot-matrix display, an image is generated by turning on some light-emitting diodes (LEDs) and turning off others, as shown in Figure 3.2.

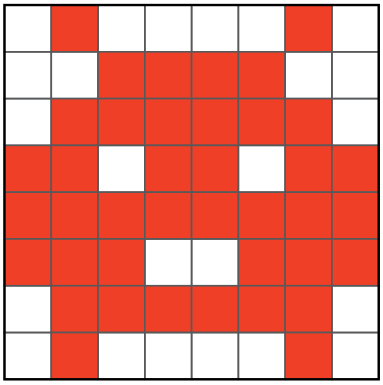


Figure 3.2 Digital image displayed on an 8x8 screen.

Moving image

A moving image is created by displaying more than one digital image consecutively, and each digital image (a frame of a moving image) is on the screen for a predefined time period before the next one is posted. For example, the first image is displayed for 0.5 second, and then the second image is displayed for 0.5 second, as shown in Figure 3.3. The time period of display should be long enough so that the human visual system can recognize the image.

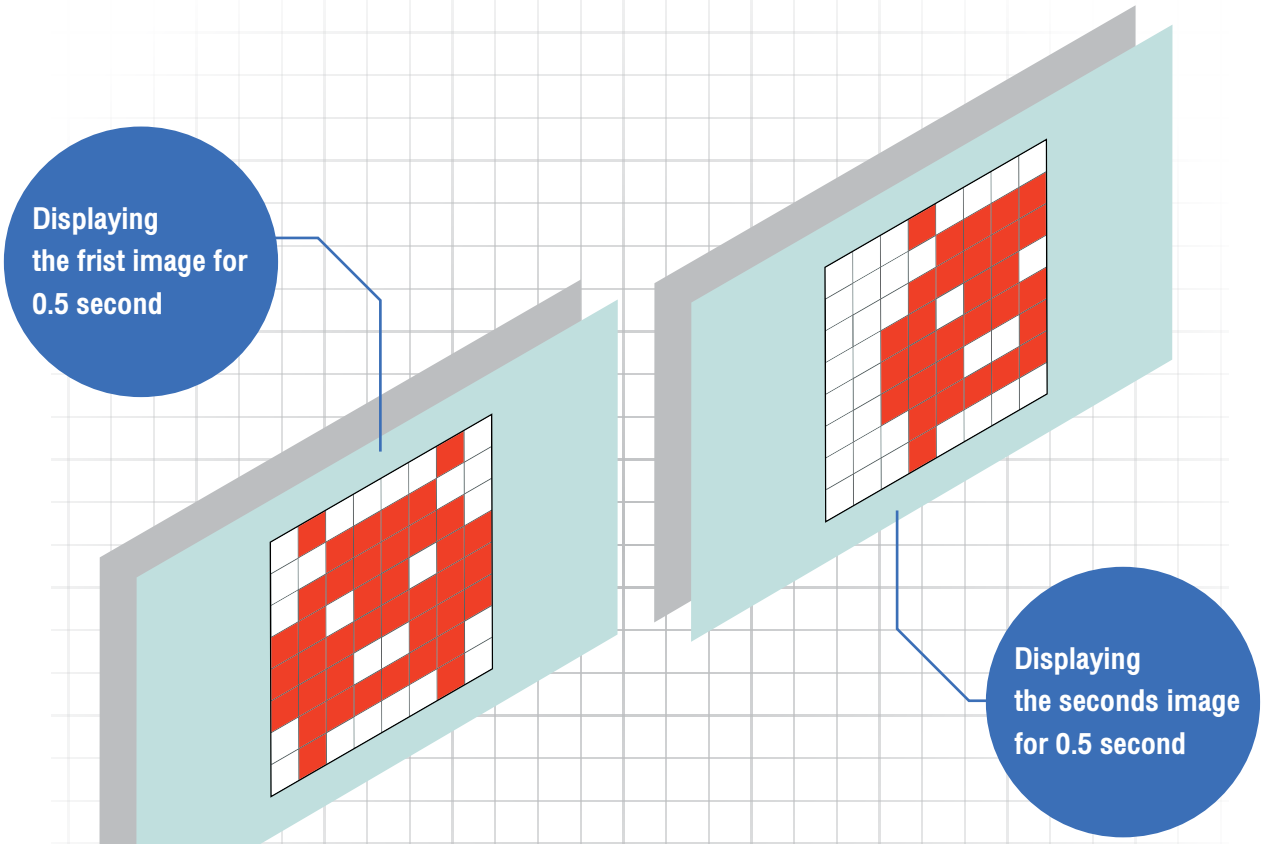


Figure 3.3 Moving image.

In Figure 3.3, it looks like the first image is moved to the right-hand side because all red pixels have moved to the right-hand side by two pixels. Adding more images produces a more complicated moving image.

The command blocks in the KidBright IDE can be used to create still and moving images.

On the Basic tab of the KidBright IDE, as shown in Figure 3.4, there are many blocks that can be used to create still images and moving images.

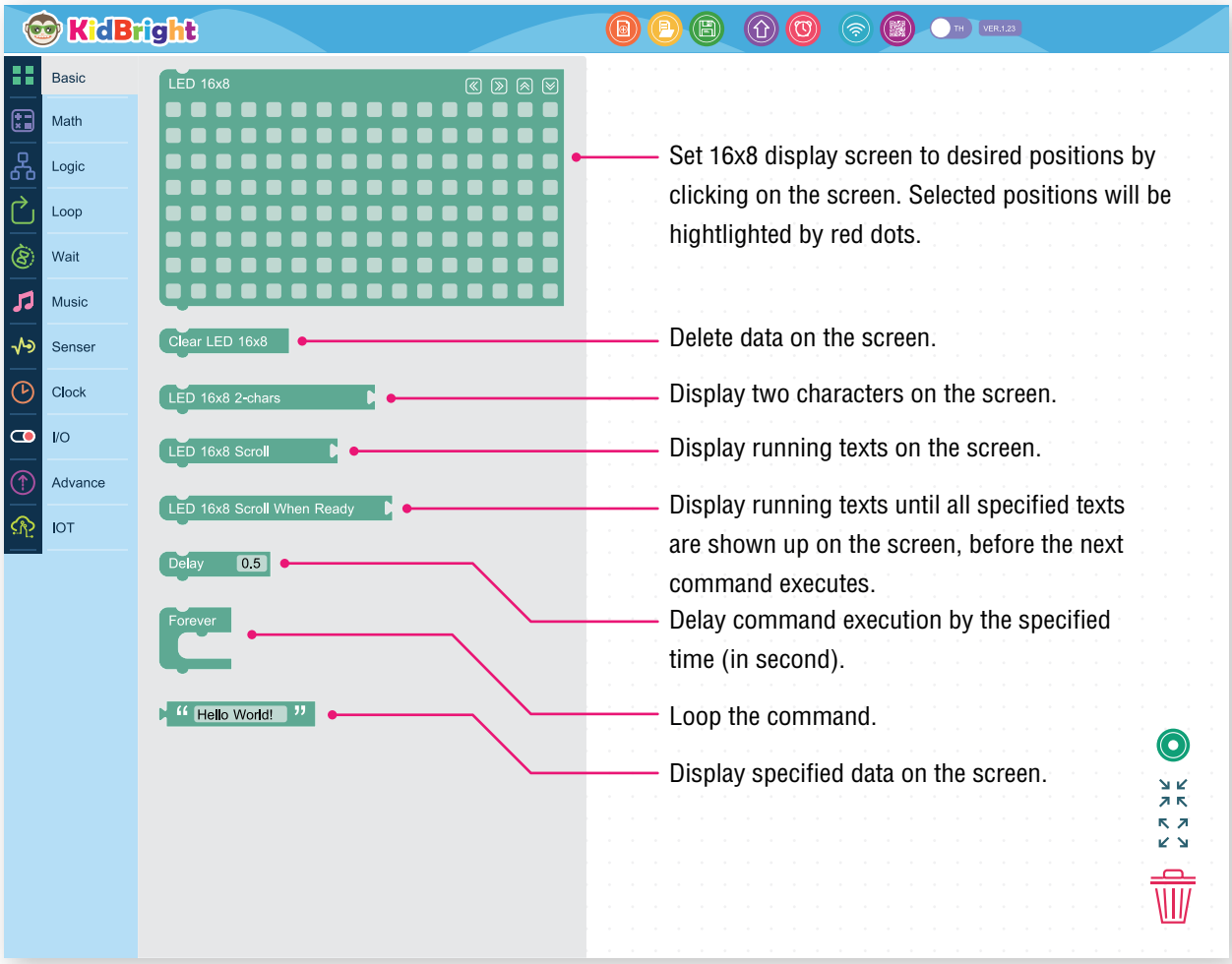


Figure 3.4 Basic tab of the KidBright IDE.

Creating a still image

In order to create a still image, the 'LED 16x8' block as shown in Figure 3.5 is used.

'LED 16x8' block

Dots in the 'LED 16x8' block represent the dot pixels of the KidBright board's screen.

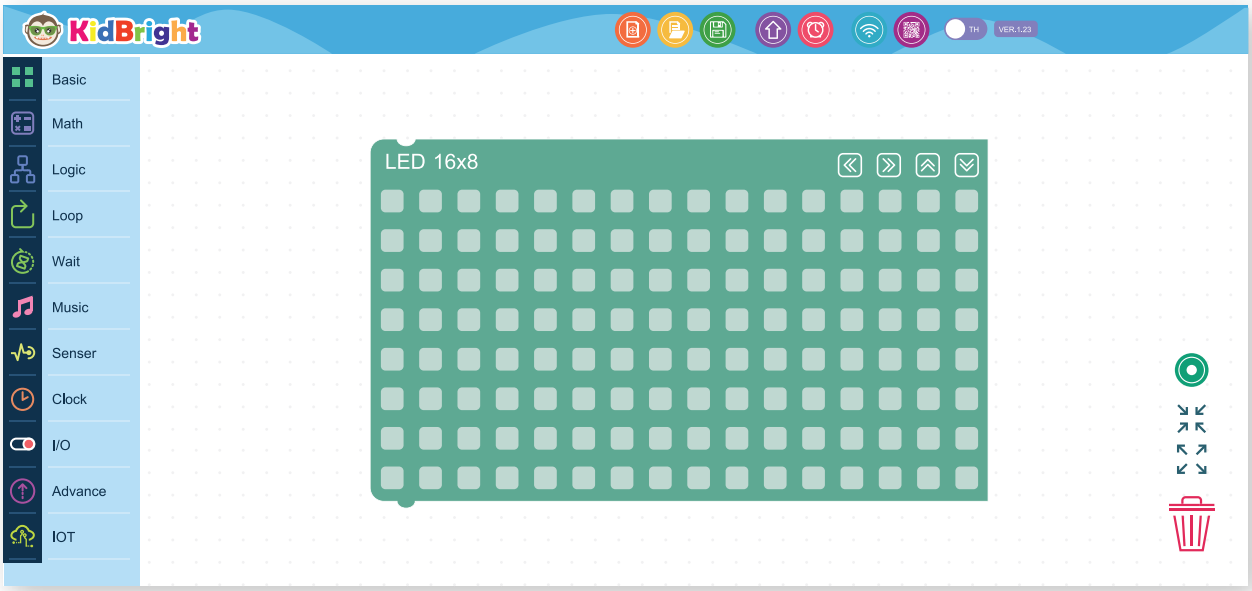


Figure 3.5 'LED 16x8' block

Activity

Activity 3.1

How to use the 'LED 16x8' block?

1. Drag the 'LED 16x8' block from the Basic tab on the left-hand side and drop it onto the working area.
2. Click on dots to be displayed as bright pixels on the KidBright screen, as shown in Figure 3.6. Once selected, the dots turn red.

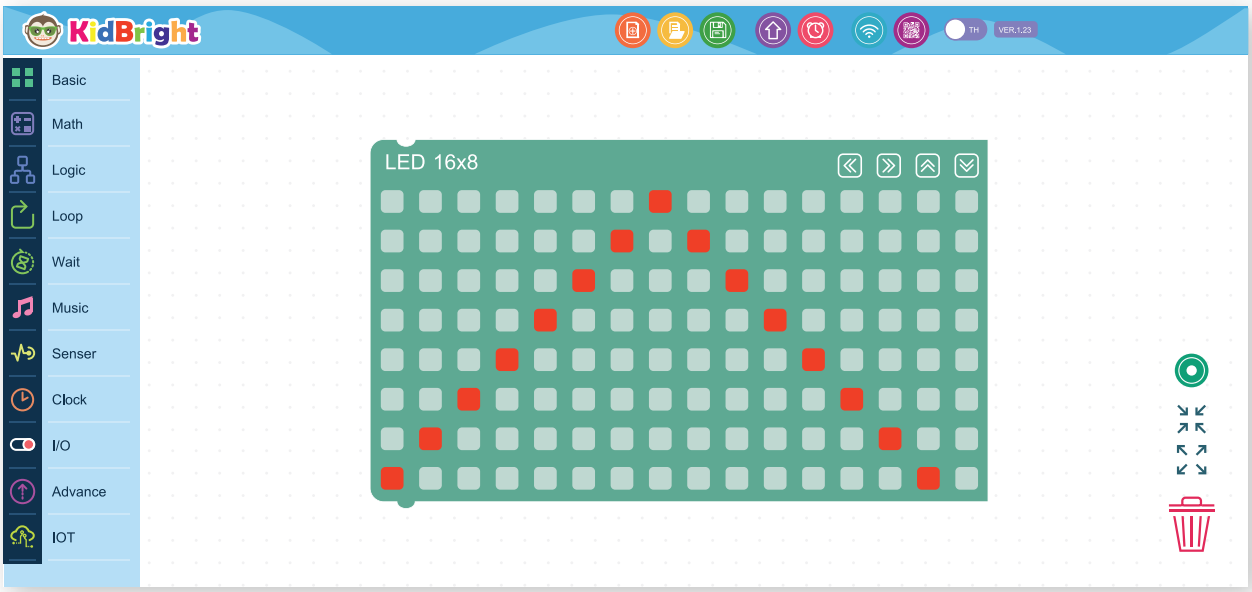


Figure 3.6 Specifying the bright pixels.

3. Click the **Program Build** button, and a window will appear to show details of the compiling process as shown in Figure 3.7.
4. Click **OK.**

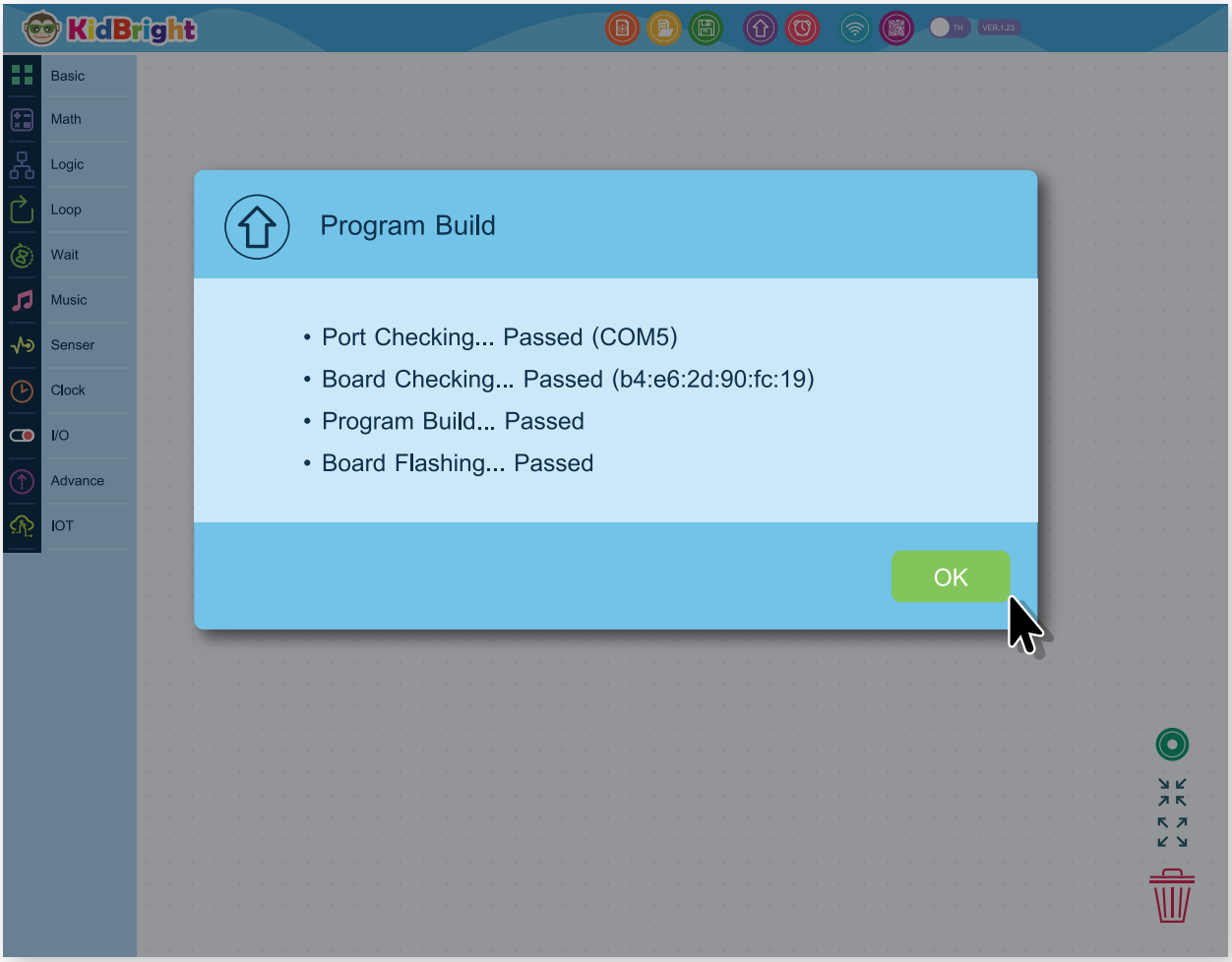


Figure 3.7 Compiling process.

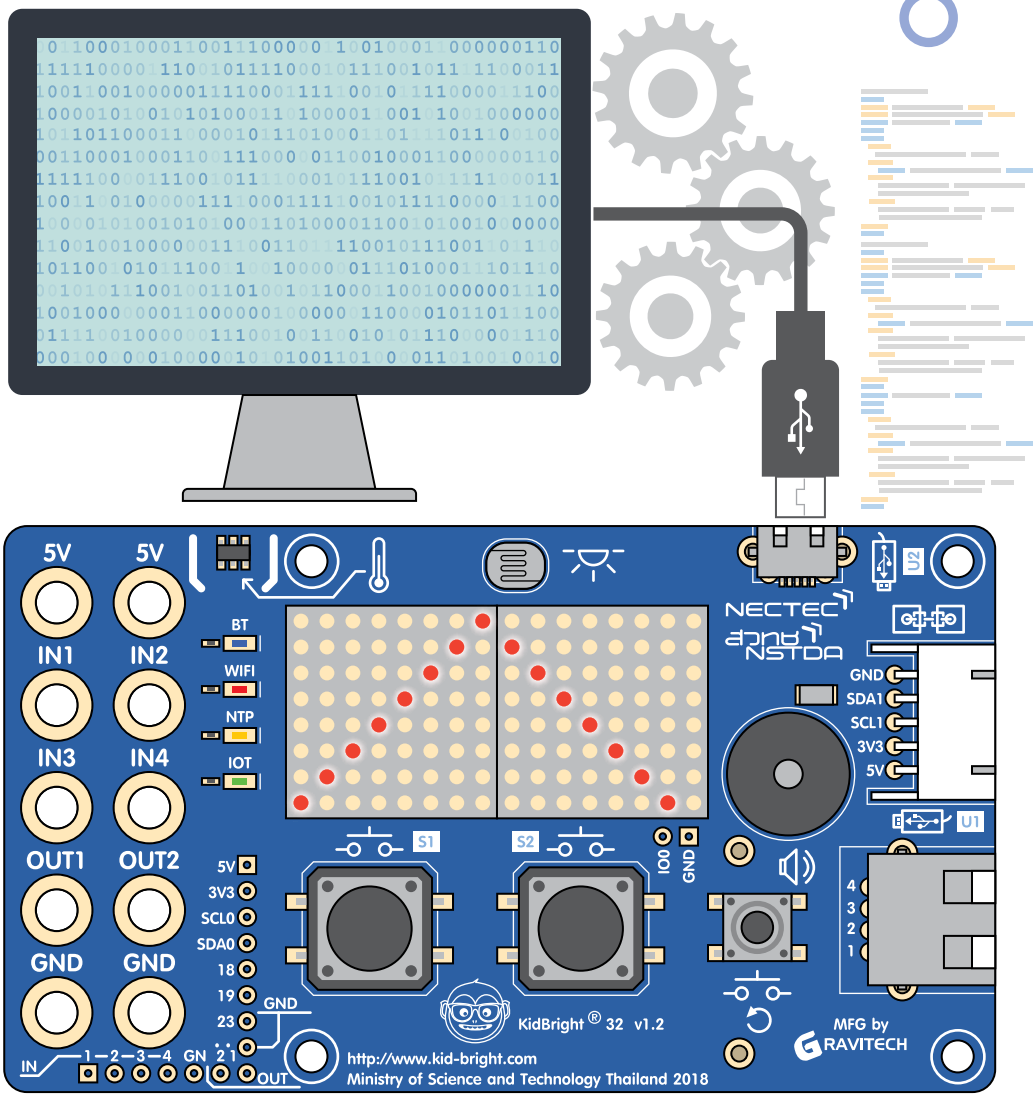


Figure 3.8 Displaying the result on the KidBright screen.

After compiling is finished, the machine code is transferred to the board via the USB connection, and the KidBright screen displays an image as shown in Figure 3.8.

To record the program (or command series), click the **Save File** button, name the file (Each OS have limitations on the file naming), and finally click the **OK** button as shown in Figure 3.9.

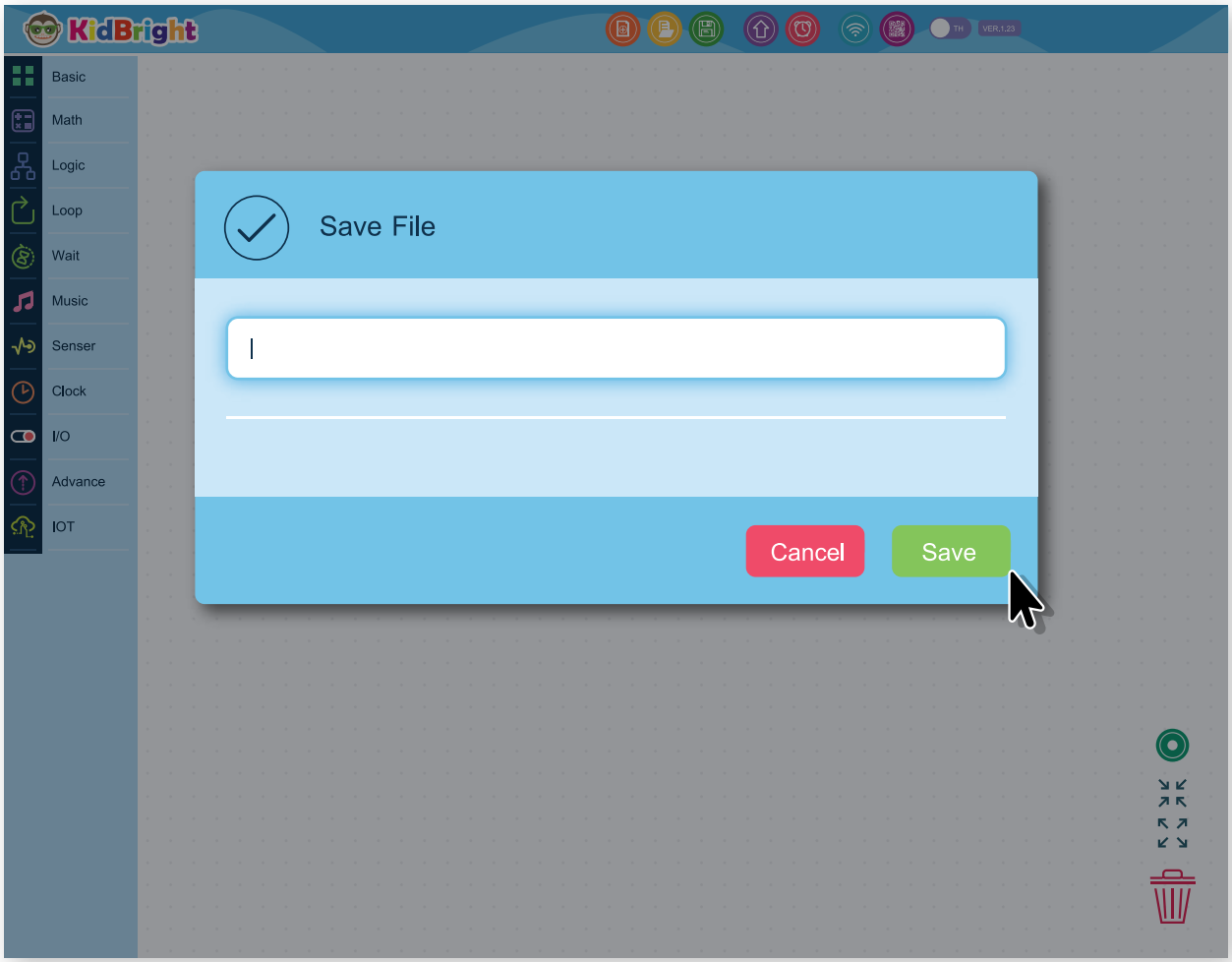


Figure 3.9 Saving the command series on a PC or a flash drive.

Creating a moving image

There are two different types of moving image: 1) moving images and 2) running texts.

1. Moving images

Commands used to create moving images are as follows:

- **'LED 16x8' block**

The 'LED 16x8' block has four support tools on the upper-right corner of the block. These tools are displayed as four arrow buttons: up, down, left, and right, as shown in Figure 3.10. These buttons can be used to move selected pixels in the desired direction so that you do not have to redraw the image, as shown in Figure 3.11.

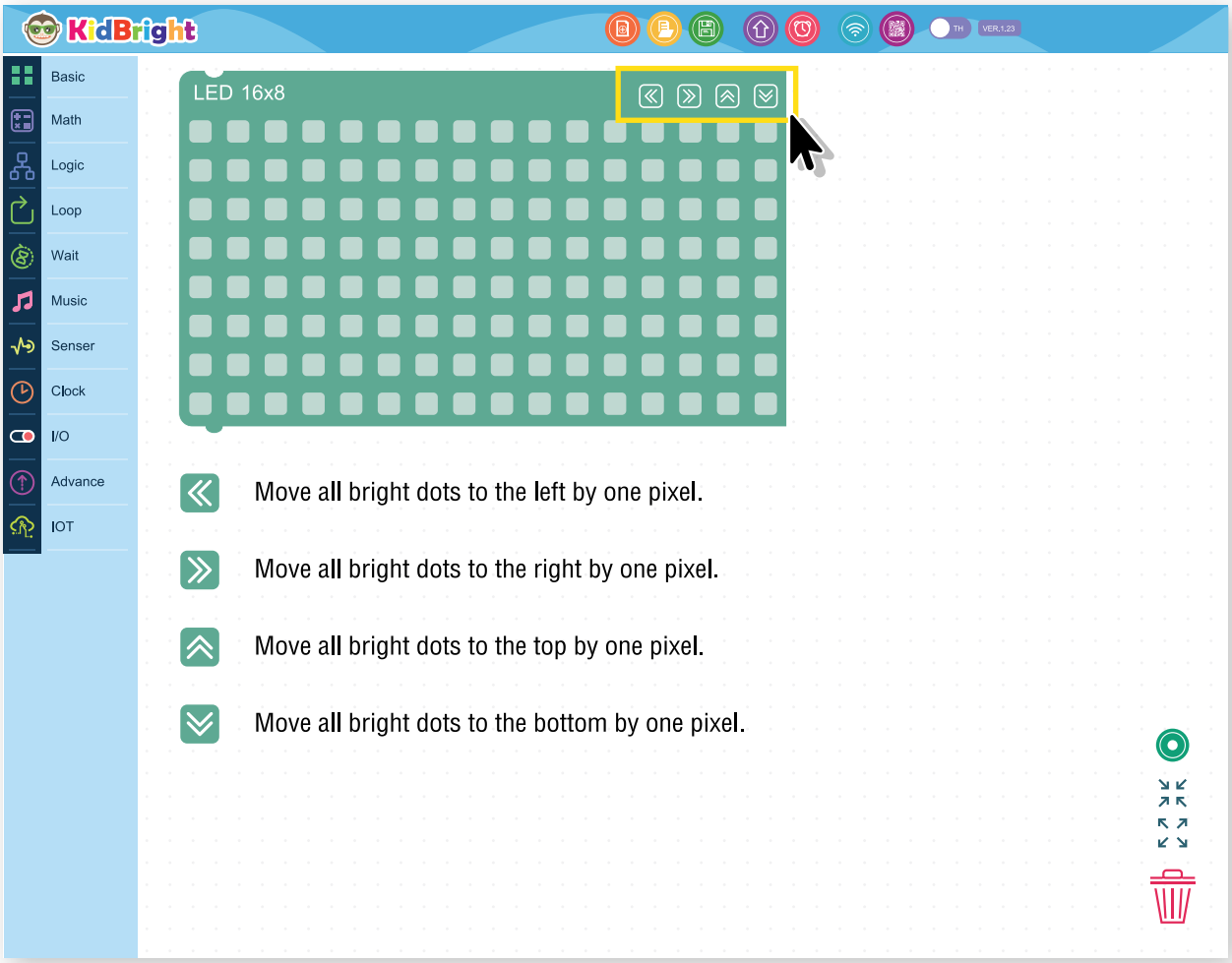


Figure 3.10 Support tools that can be used to move the bright dots on the 'LED 16x8' block.

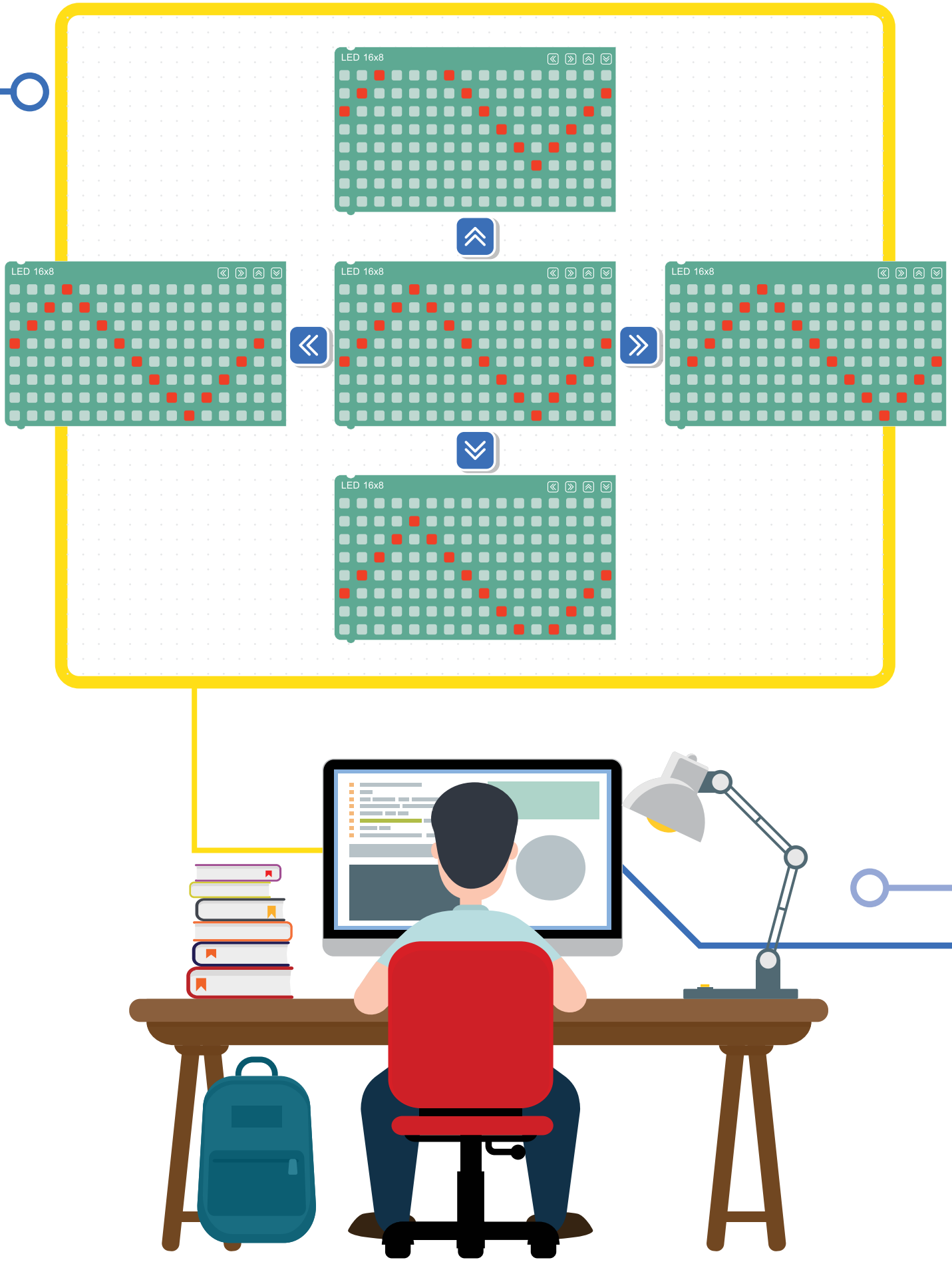


Figure 3.11 Images obtained after using the arrow buttons.

- **'Delay' block**

The 'Delay' block can be used to command the microcontroller to temporally pause for a specified time period before executing the next commands. The time value to be set is in a unit "milliseconds". Note that 1000 milliseconds are equal to 1 second.

Figure 3.12 shows the 'Delay' block that is used to freeze the screen for a specified duration so that the human visual system can recognize an image.

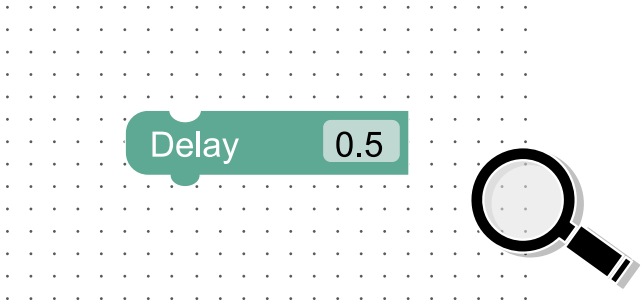


Figure 3.12 'Delay' block.

- **'Forever' block**

A loop (or a repetition) is a command that asks the microcontroller to repeatedly execute commands under it until a given condition is satisfied. There are two types of loop: 1) Unconditional loop (e.g., the 'Forever' block) and 2) Conditional loop i.e., a loop that will end when a specific condition is met.

Figure 3.13 'Forever' block.

!

When you want to display a continuous moving image, you use the 'Forever' block on the Basic tab. The image will be shown continuously (or until the program is stopped).

Activity 3.2

Write a program to display a sailing ship moving across the screen by using the left and the right arrow on the ‘LED 16x8’ block together with the ‘Delay’ and ‘Forever’ blocks.

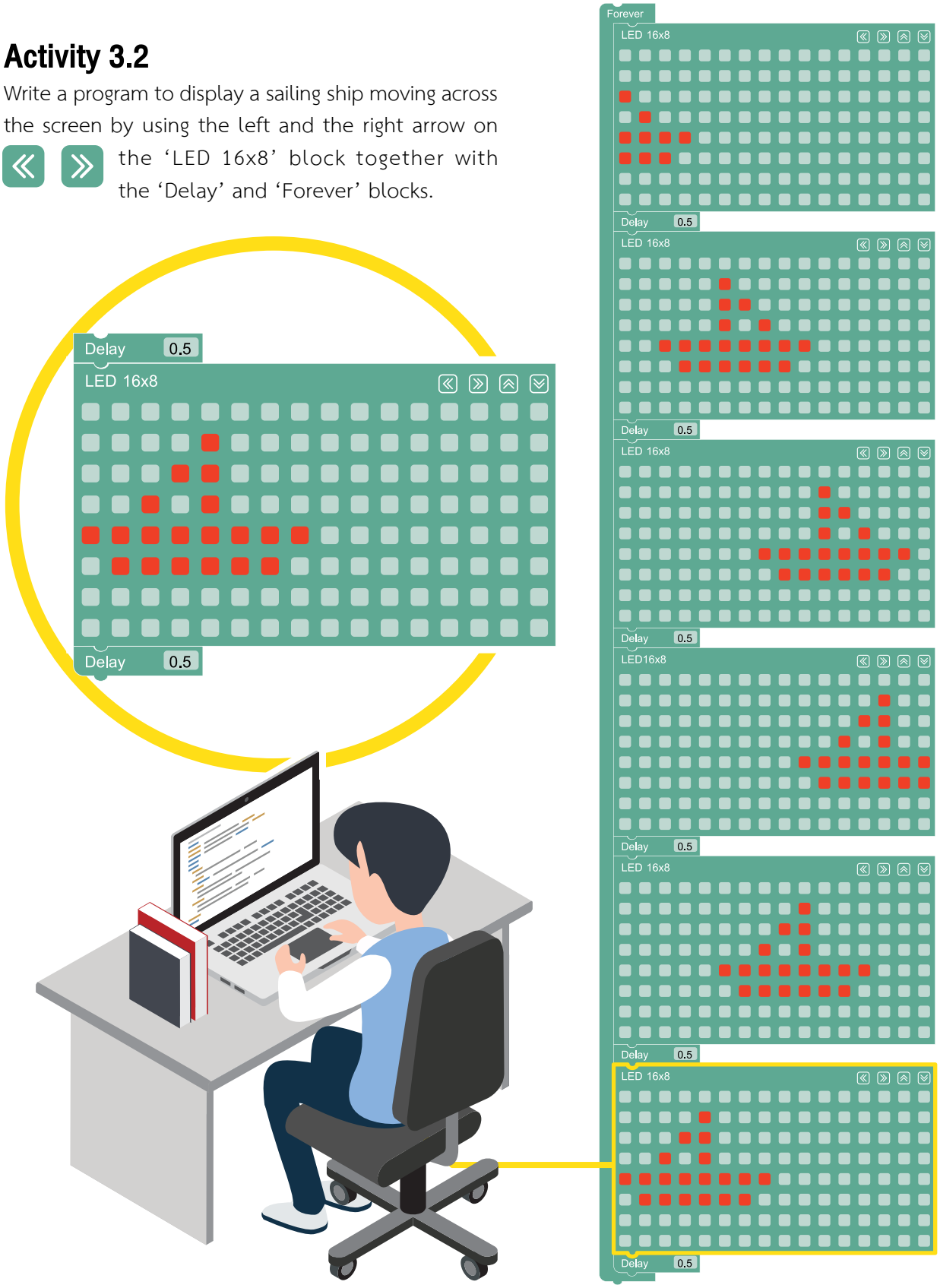


Figure 3.14 Example of writing a program to create an image of a sailing ship that will reverse its direction every time it reaches the end.

Activity 3.3

Write a program to show a moving image in the left-right and up-down directions.

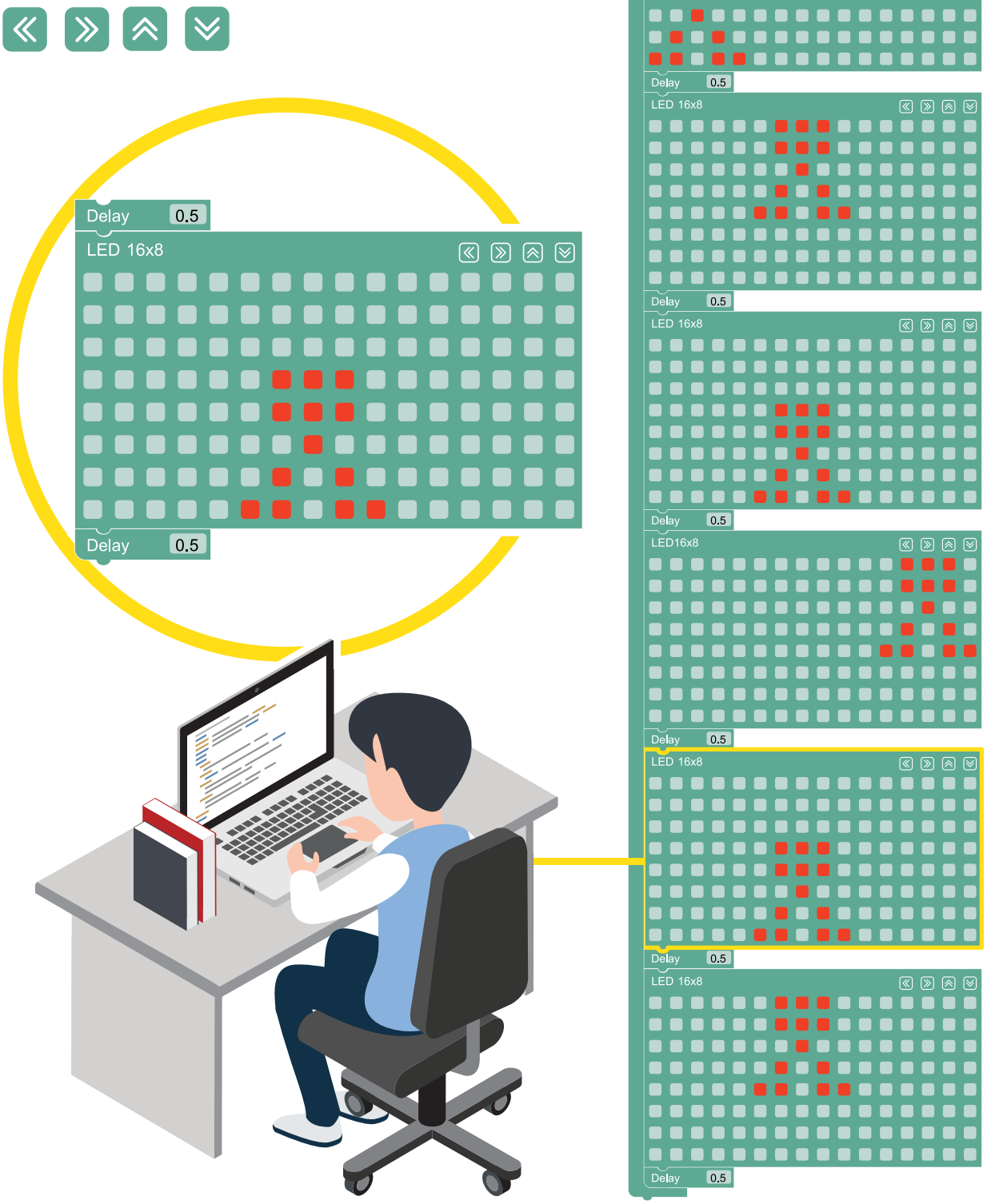


Figure 3.15 Example of the program that displays a moving image in left-right and up-down directions.

2. Running text

The KidBright screen is a 16x8 dot matrix which can show only two text characters (one character per 8x8 dot-matrix). If you want to display more than two characters, the characters must be shifted to the left to allow a new character to appear on the screen.

The KidBright IDE provides three blocks for this task as follows:

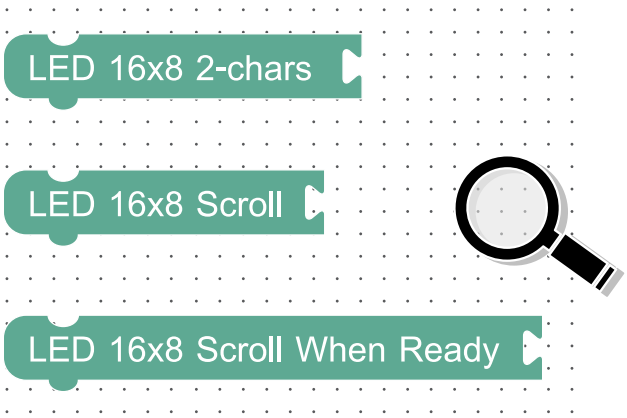


Figure 3.16 Character/text display blocks

> Three character/text blocks (Figure 3.16) can work when putting a 'Text' block (Figure 3.17) after them.

'LED 16x8 2-chars' block

This block displays only two characters without running of these characters on the screen.

'LED 16x8 Scroll' block

This block displays a text of more than two characters running slowly from right to left until the last character is displayed. Note that using this block to overwrite anything display on the screen including another "LED 16x8 Scroll block, therefore, we must use the 'Delay' block for showing text for a specified time period before the next command executed.

'LED 16x8 Scroll When Ready' block

This block displays a text running slowly to the left-hand side until the last character is on the screen. It will wait until all specified texts are shown up on the screen before the next command executed.

'Text' block

This block allows users to put a text to be displayed on the screen. Note that the maximum number of the displayed characters is 31. Figure 3.17 shows "Hello World!" using a 'Text' block.

Figure 3.17 'Text' block

Activity 3.4

Write a program to show “Hello World” on the screen as shown in Figure 3.18.

Figure 3.18 Program that displays “Hello World” on the screen.

The result after compiling is shown in Figure 3.19.

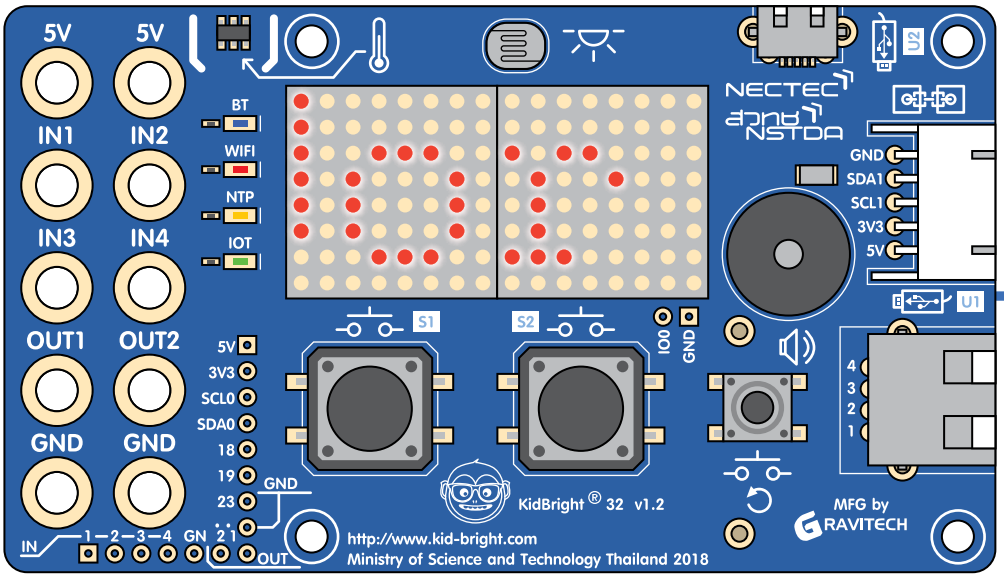
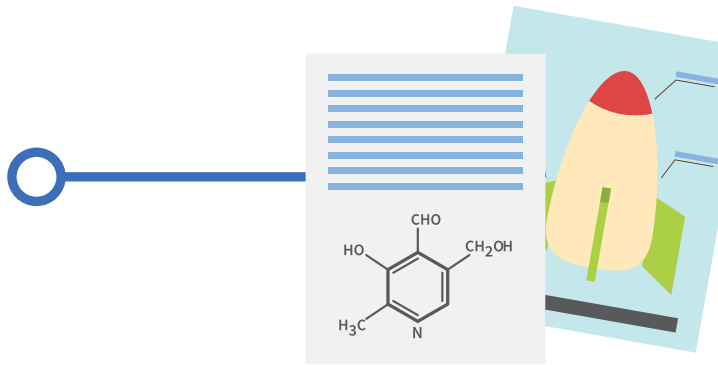


Figure 3.19 “Hello World” is displayed on the screen.

!

Students are expected to understand how to create still images and moving images, and use blocks in the KidBright IDE to produce images.

Exercise



1. Create an image from your imagination using blocks on the Basic tab.

2. Show any text on the KidBright Board's screen using blocks on the Basic tab.

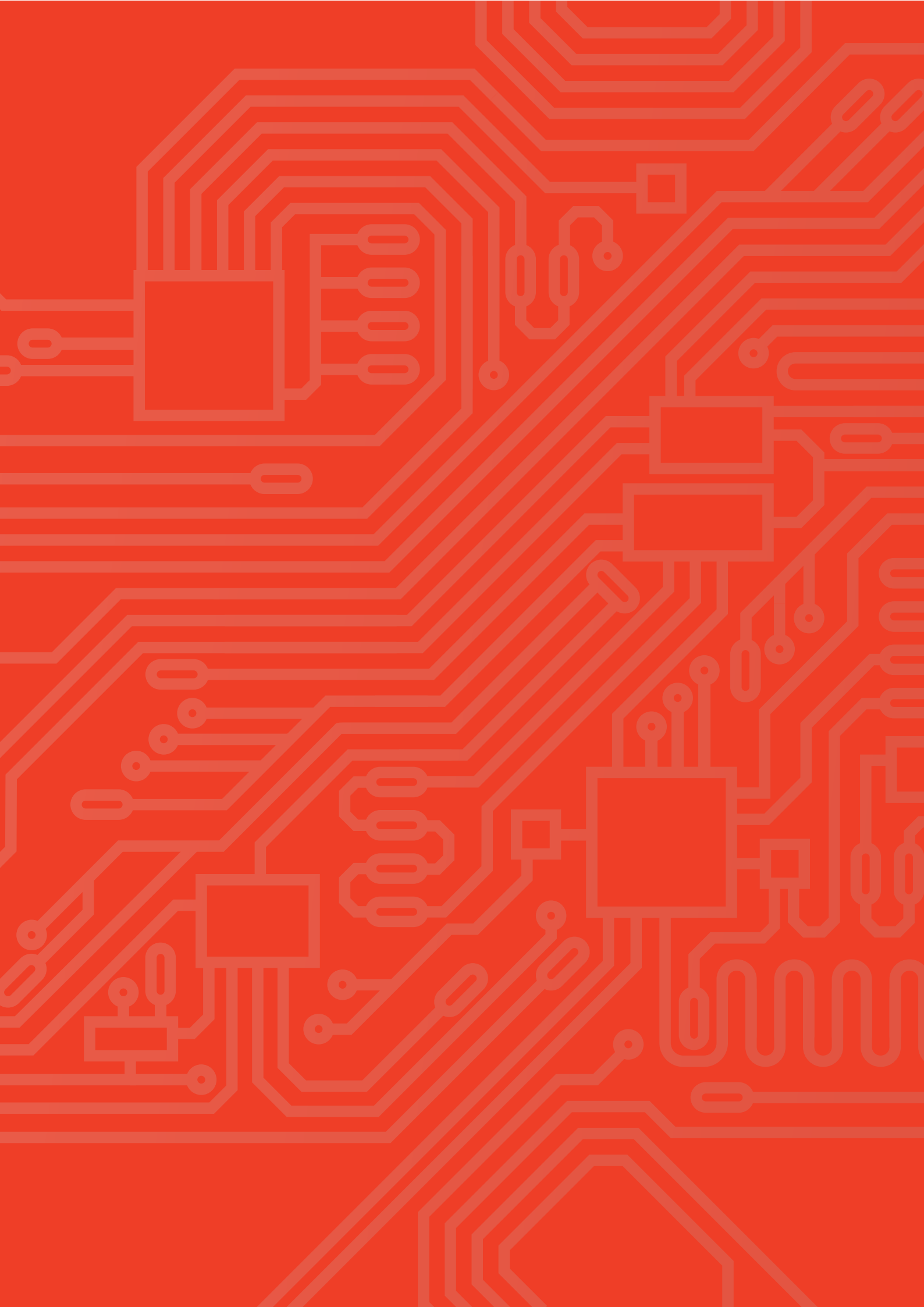
Chapter 4

Mathematics with KidBright

Learning Objectives

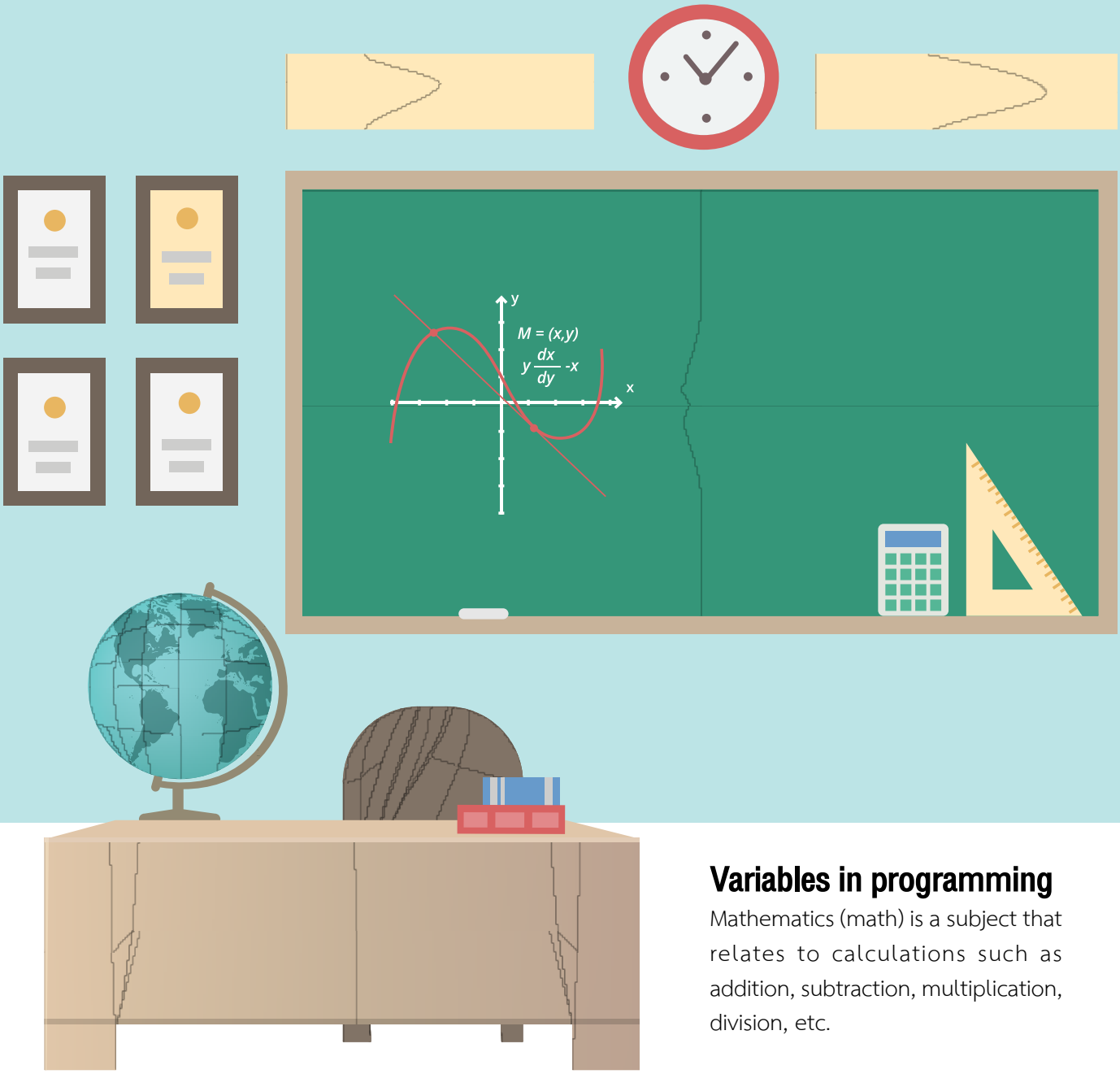
At the end of the learning process of Chapter 4, students are able to:

1. Create and use variables.
2. Set values to variables.
3. Create commend series for mathematical computation.



Chapter 4

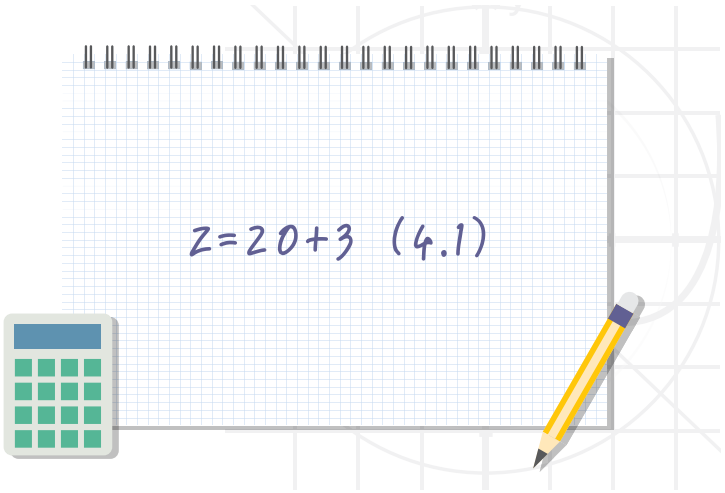
Learning Content



Variables in programming

Mathematics (math) is a subject that relates to calculations such as addition, subtraction, multiplication, division, etc.

The result of calculation comes from an operation involving at least two numbers, for example:



The above figure shows the value of Z derived from two positive integers, (20 and 3) that are added together. (4.1) is called an “equation”.

In computer programming, (4.1) is done by adding 20 with 3 and then storing the obtained result to Z. Z is called a “variable”. It is important to understand how variables are used for mathematical calculation in programming no matter what programming language is used.

In this chapter, we focus on variables and mathematical calculations of variables. Students will know how to create variables, set variables, and use variables in mathematical calculations, for example, how to calculate the area or a volume of various objects.

Mathematics block

In chapter 2, expoloring KidBright world, we already learned that there are a number of mathematic block commands in KidBright IDE that are grouped together in Math tab, as shown in Figure 4.1.

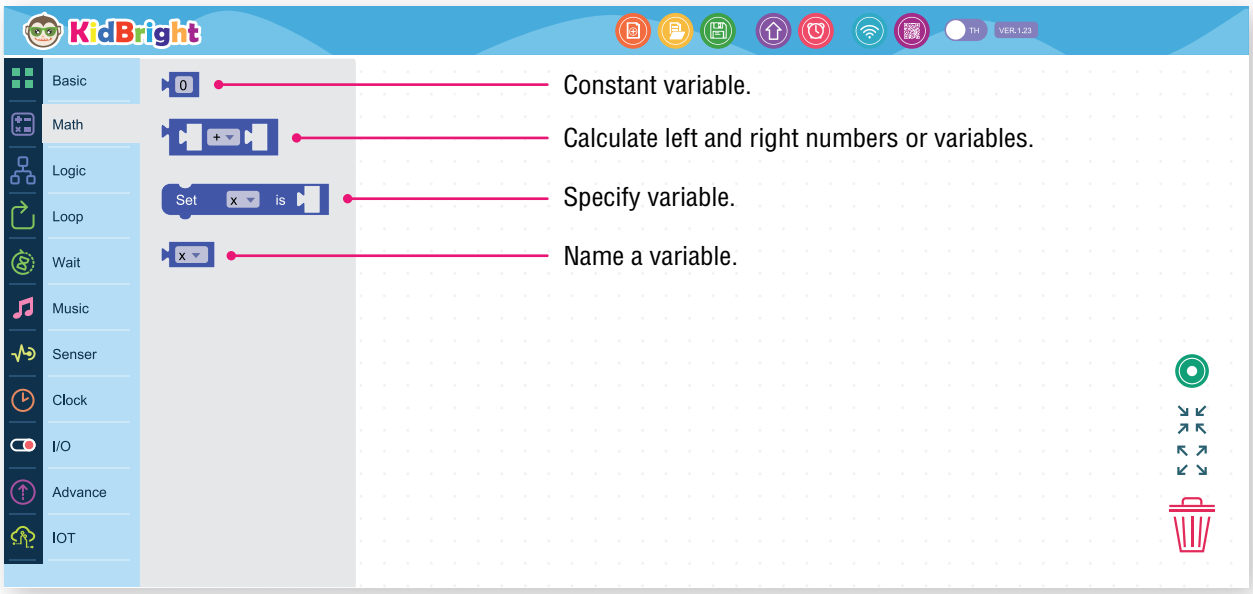


Figure 4.1 Command blocks in the Math Tab.

'Constant' block

This block is used to set a constant value, i.e. a value that does not change. Users can specify the required value in the block by typing the number on a computer keyboard. This specified value can be an integer or a decimal.

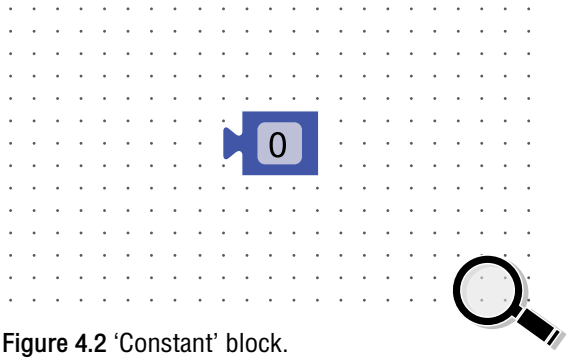


Figure 4.2 'Constant' block.

‘Mathematical calculation’ block

This block is used for mathematical calculation such as addition, subtraction, multiplication, division, and modulo (the remainder after division). The symbol ‘%’ is used for the modulo operation. Here is an example for modulo calculation: the division of a positive integr 10 by a positive integer 2 can be written as 10%2. The obtained result is equal to 0 (because 10 divided by 2 is 5 without any remainder).

Figure 4.3 ‘Mathematical calculation’ block.

‘Variable’ block

This block is used to set up desirable variables. Creating a variable simply begins with the naming of the variable. X is set as a default variable name in KidBright IDE. In case that users want to define variables with other names, it can be done by clicking on the down arrow and a dropdown message box will show up as shown in Figure 4.4. By clicking **“Rename Variable...”**, a rename variable box will show up as shown in Figure 4.5. Users name for the variable (variable name should not be reserved words in C programming language). In this example, the new variable is set as Z. Then click on **OK** button after changing the name. The new variable name will show up in the command and ready to be used.

Figure 4.4 ‘Variable’ block

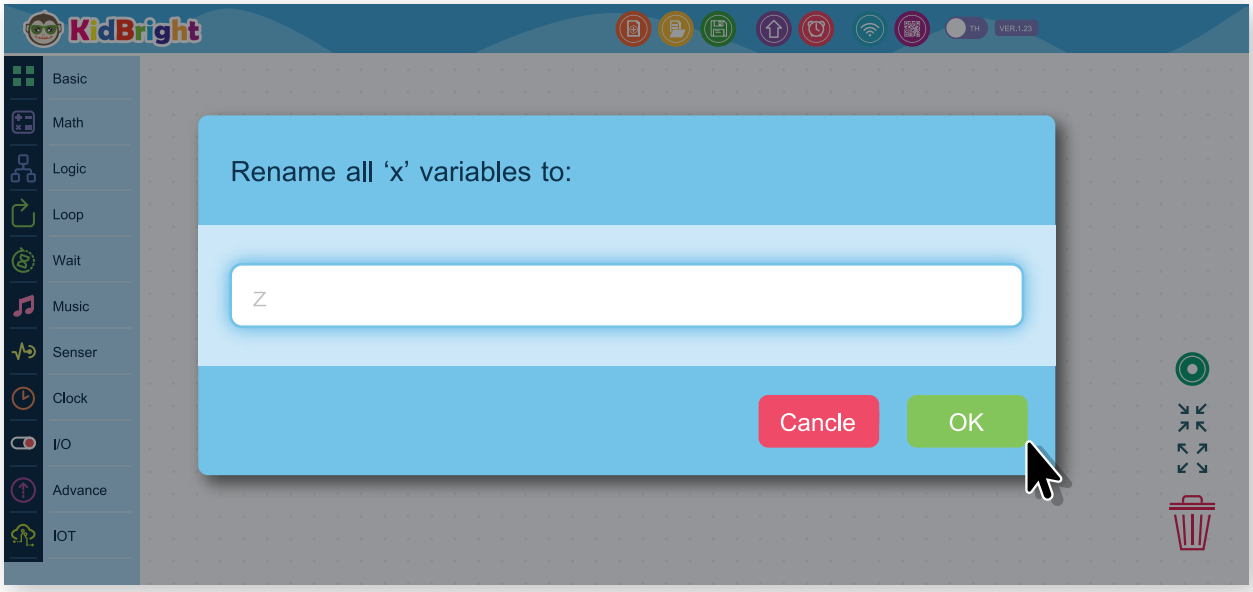


Figure 4.5 Changing variable name box.

‘Variable setting’ block

This block is used for setting value for a variable. Users can set the value by clicking at the dropdown arrow. Variables set in the program will appear. Users then select a desirable variable as shown in Figure 4.6. However, all variables need to be created with ‘Variable’ block before they appear in the variable selecting list.

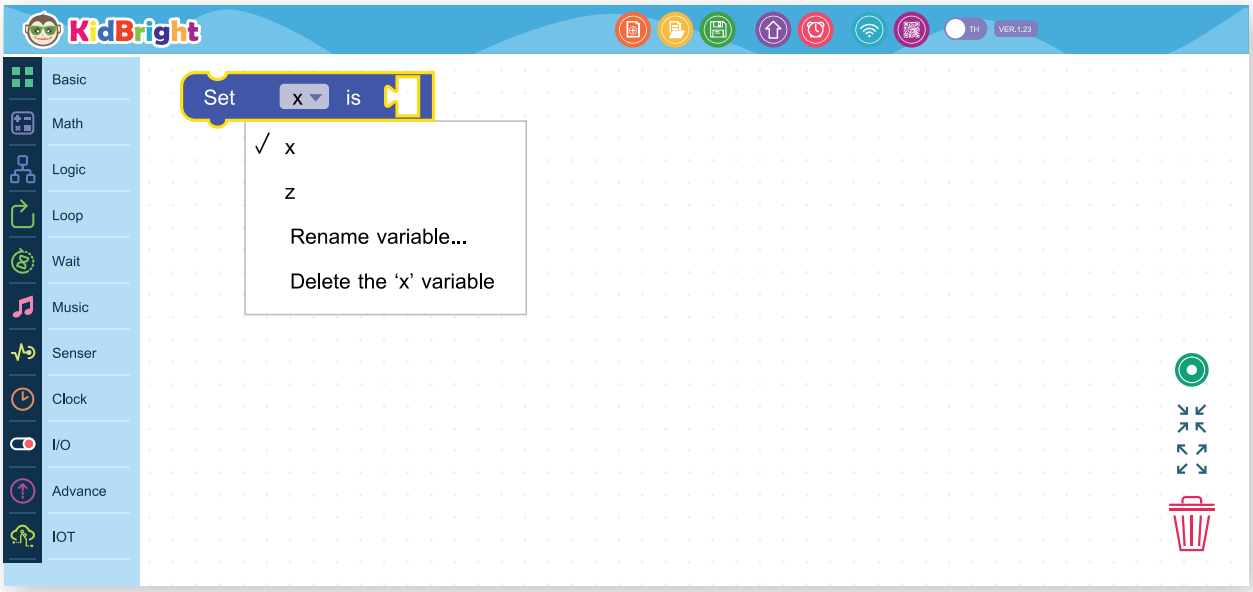


Figure 4.6 ‘Variable setting’ block.

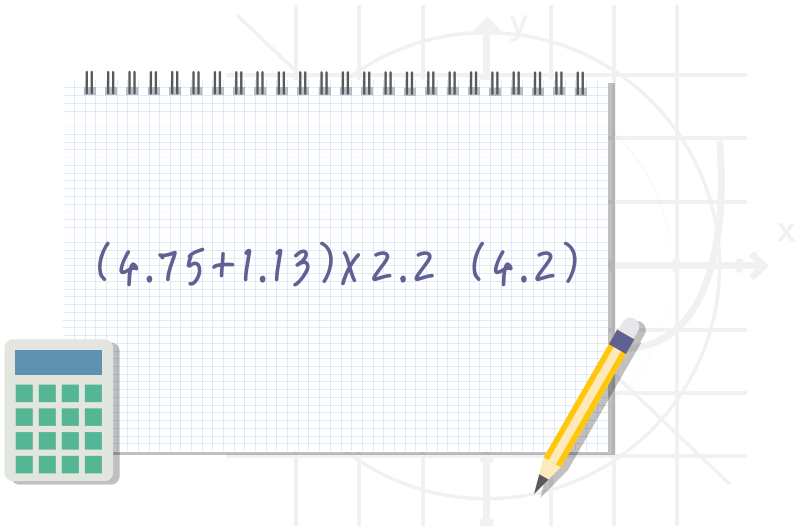
Activity

Activity 4.1

Mathematical calculation

In KidBright IDE, the Math tab has many command blocks for mathematical calculations such as addition, subtraction, multiplication, division and modulo for computing the remainder after division.

Using the blocks in KidBright IDE to find the answer for equation 4.2 as follows:



1. In KidBright IDE, drag a 'LED 16x8 Scroll When Ready' block to the programming area, then attach the 'Mathematical calculation' block to the end.
2. Click the dropdown arrow inside the 'Mathematical calculation' block to select the multiplication symbol.
3. In the first blank slot of the 'Mathematical calculation' block, place the second 'Mathematical calculation' block inside the first one.
4. Click the dropdown arrow inside the second 'Mathematical calculation' block to select the addition sign.
5. Drag three 'Constant' blocks and set the values to 4.75, 1.13, and 2.2, respectively. Then place them into the blank three slots of the dragged 'Mathematical calculation' blocks as shown in Figure 4.7.
6. Click the **Program Build** button to create the program. When it is finished, the board will display the calculated result as shown in Figure 4.8.

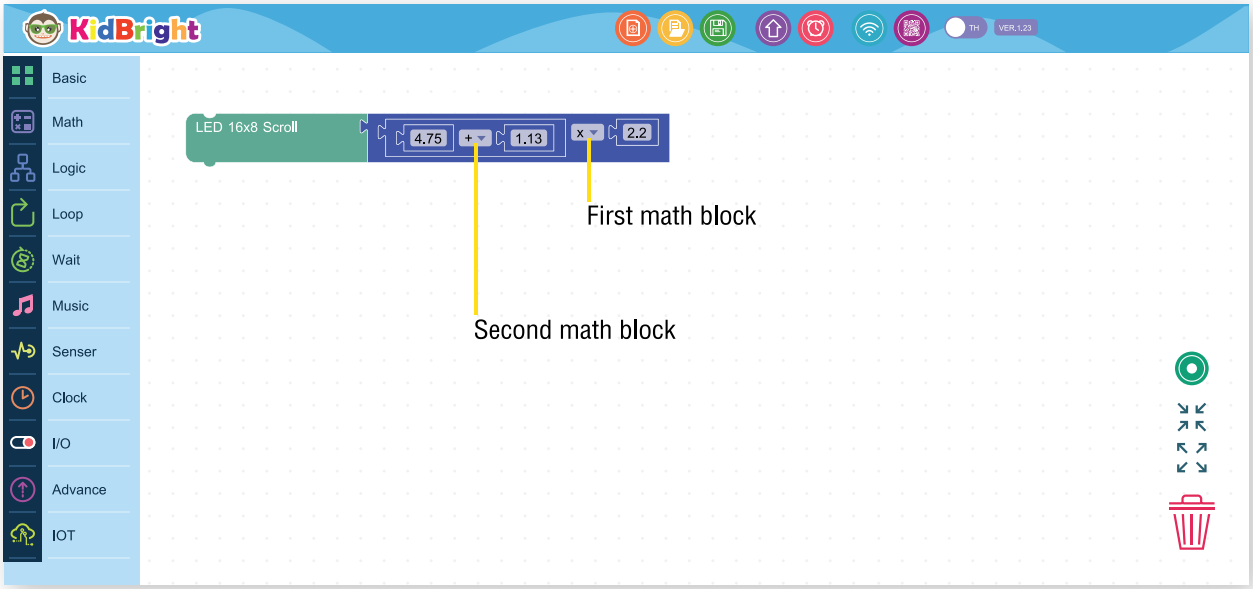


Figure 4.7 Mathematical calculation program.

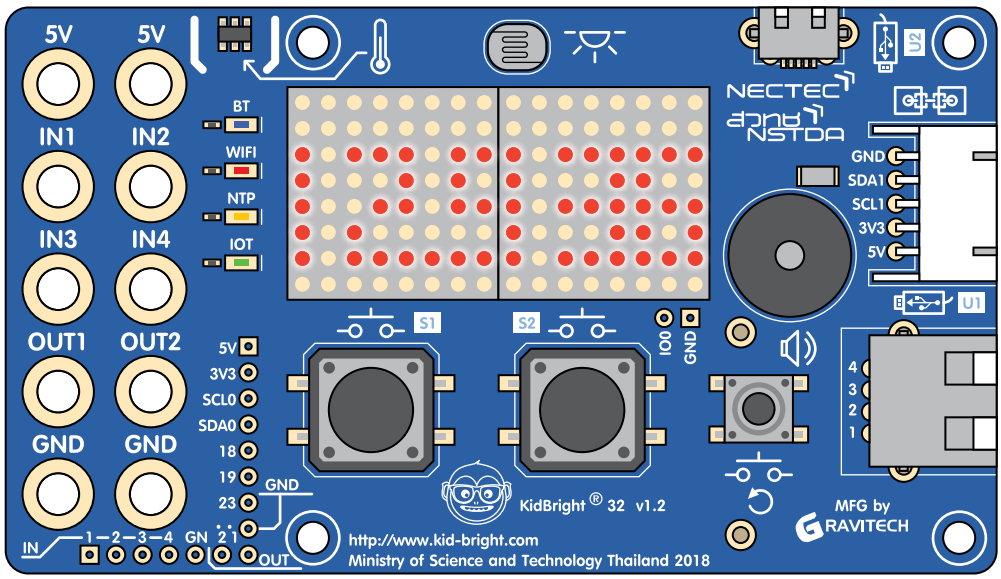
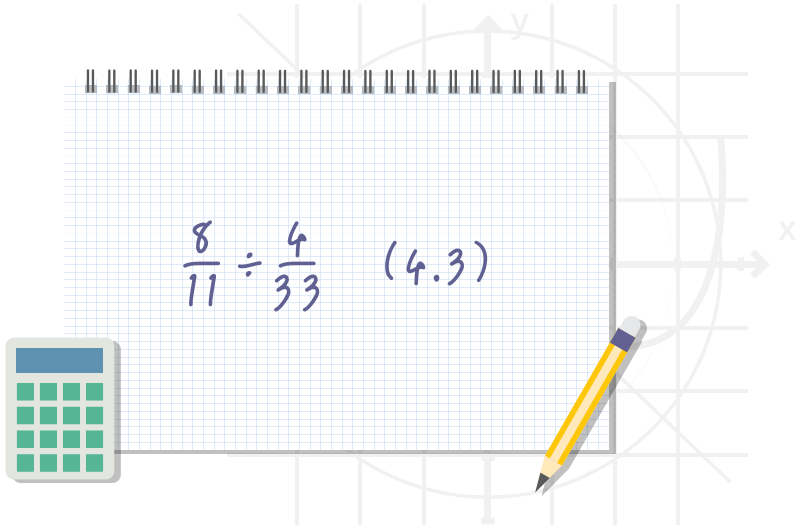


Figure 4.8 Mathematical calculation result.

Activity 4.2

Fraction calculation

Using KidBright IDE to find the answer for equation 4.3.



1. In KidBright IDE, drag a 'LED 16x8 Scroll When Ready' block to the programming area, then attach 'Mathematical calculation' block to the end.
2. Click the dropdown arrow inside the 'Mathematical calculation' block to select a division symbol.
3. In the first blank slot of the first 'Mathematical calculation' block, place the second 'Mathematical calculation' block inside the first one.
4. Click the dropdown arrow inside the second 'Mathematical calculation' block to select a division sign.
5. In the second blank slot of the first 'Mathematical calculation' block, place the third 'Mathematical calculation' block inside.
6. Click the dropdown arrow inside the third 'Mathematical calculation' block to select a division sign.
7. Drag four 'Constant' blocks and set the values to 8, 11, 4, and 33 respectively, as shown in Figure 4.9.
8. Click the **Program Build** button to create the program. When it is finished, the board will display the calculated result as shown in Figure 4.10.

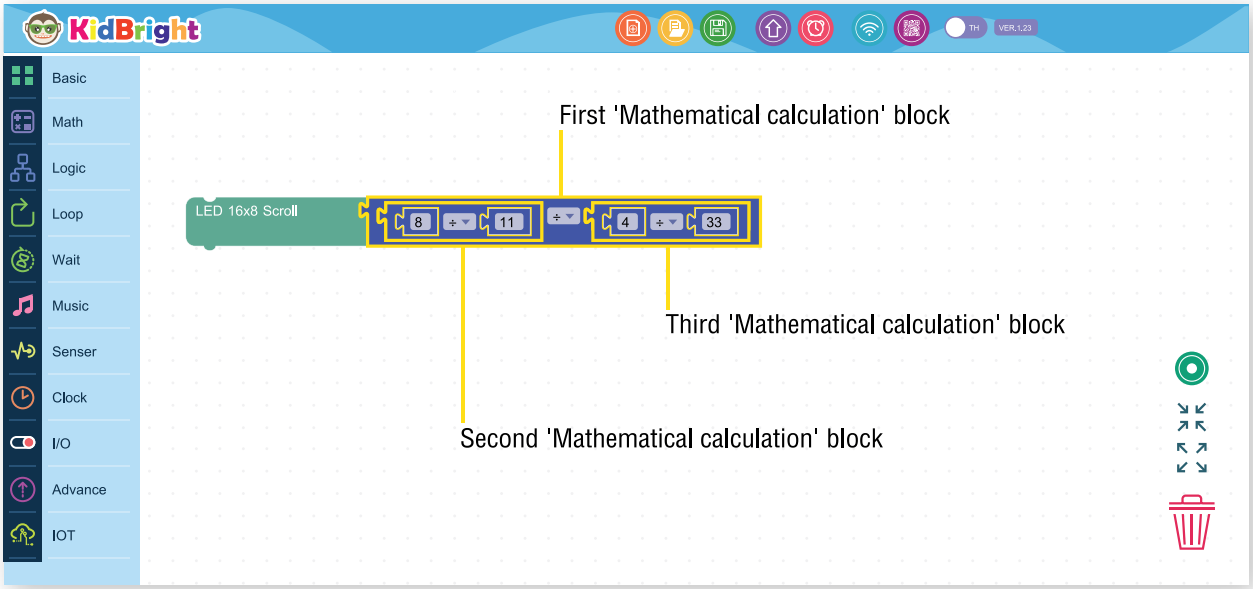


Figure 4.9 Fraction calculation program.

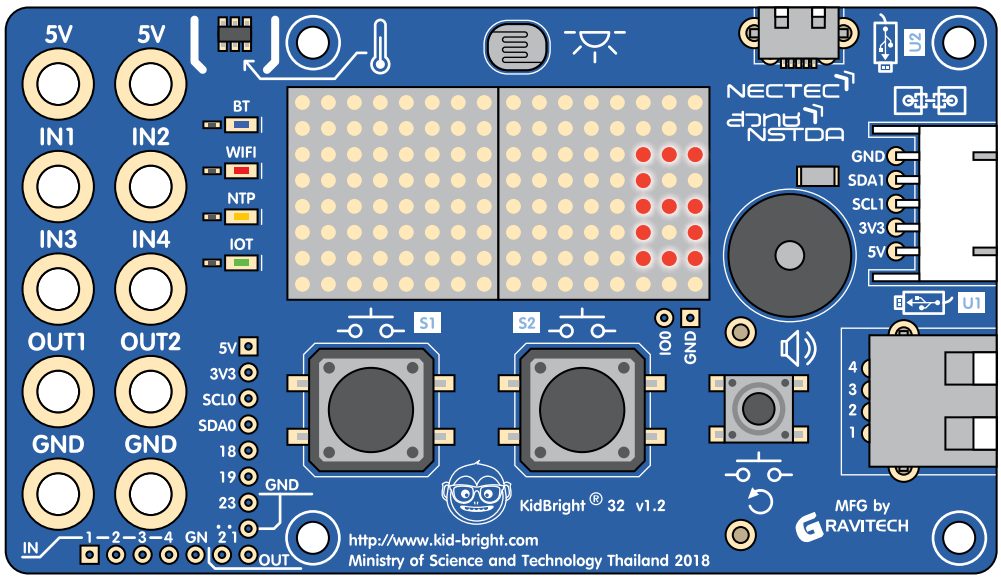


Figure 4.10 Result of the fraction calculation.

Activity 4.3

Using a variable

The important role of variable in programming is to keep the constant or different types of number to be used in mathematical calculations. Constant values are real numbers including integers (positive, negative and zero) and decimal numbers.

Procedure of using a variable to keep a constant value

1. On Math Tab in KidBright IDE, drag the 'Variable setting' block into the programming area. Then, place the setting 'Constant' block to the end of the 'Variable setting' block before changing the value of the variable to be -10.
2. Drag the 'Variable' block and connect it at the end of the 'LED 16x8 Scroll When Ready' block by choosing the variable name to be x, as shown in Figure 4.11.

Figure 4.11 Setting a value to a variable.

3. Click the **Program Build** button. When the execution is ready, the board will display the value on the screen as shown in Figure 4.12.

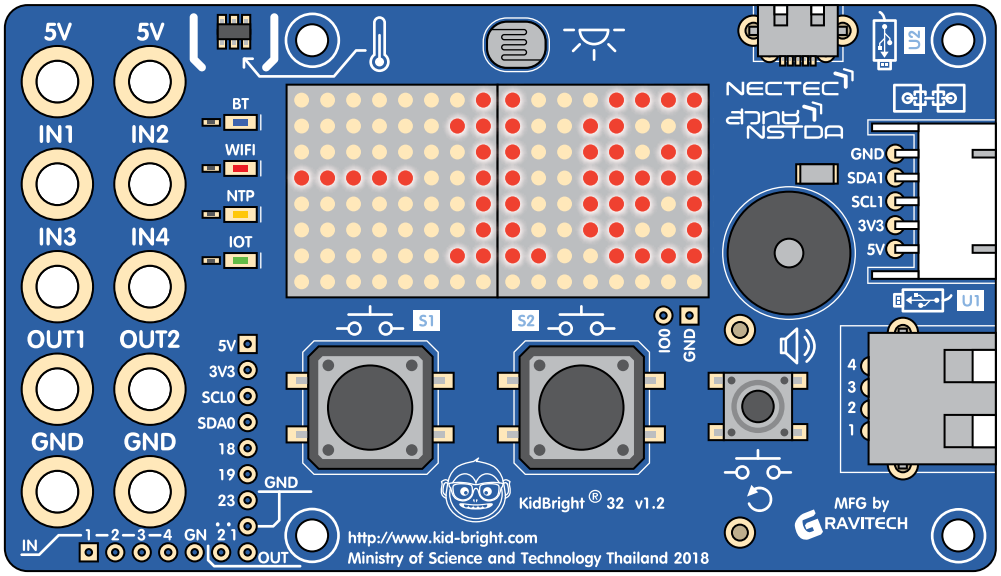


Figure 4.12 The outcome of setting a value to a variable shown on the screen.

4. From this example, try to change values of the variable to other numerals such as 10, 0.01, 10.5 and then observe the outcome.

Activity 4.4

Finding an area of a triangle

To find an area of a triangle as shown in Figure 4.13, we must know the base length and the height of the triangle to calculate the area by using the following formulae.

$$\text{Triangle area} = \frac{1}{2} \times \text{Base length} \times \text{Height} \quad (4.4) \text{ or in short as}$$

$$\text{Triangle area} = \frac{1}{2} \times \text{Base} \times \text{Height} \quad (4.5)$$

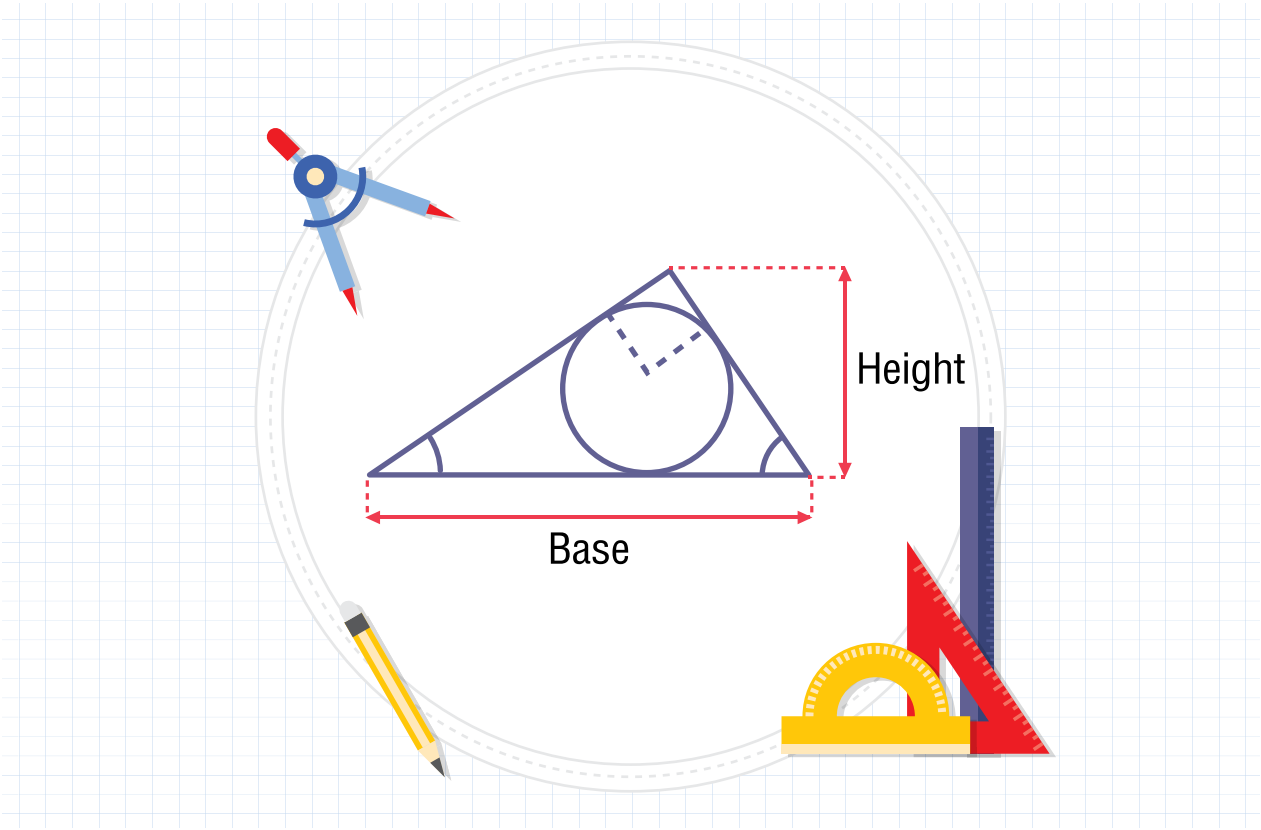


Figure 4.13 Height and base length of a triangle.

To calculate the area of a triangle by using KidBright, it can be done by using the 'Mathematical calculation' blocks in KidBright IDE as follows:

1. Create three different variables named **Base, Height,** and **Area** to represent the base length, height, and area of the triangle, respectively, by using 'Variable' block command.
2. Set the values to the **Base** and **Height** variables by assigning the Base value to be 30 and the Height value to be 10.
3. From equation (4.5), set the value to the variable **Area** by creating a set of command blocks as shown in Figure 4.14.

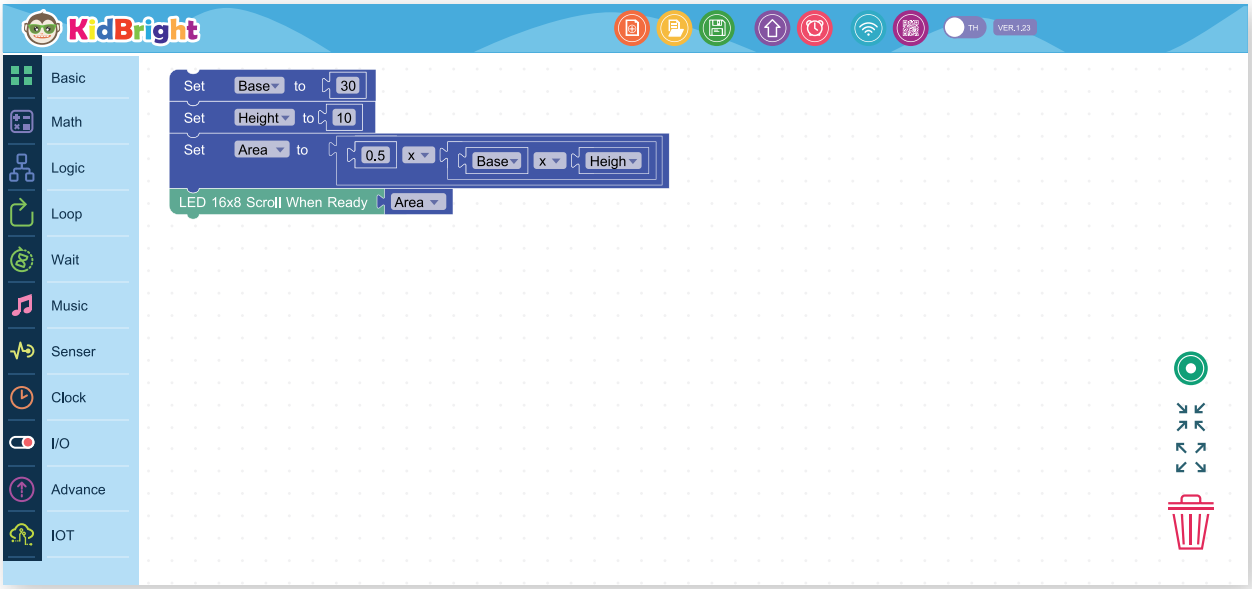


Figure 4.14 Command blocks for showing the calculation result of assigning values to variables.

From Figure 4.14, there are three variables as:

Base - For storing the value of triangle's base length

Height - For storing the value of triangle's height

Area - For storing the value of triangle's area

From the equation (4.5), we replace $\frac{1}{2}$ with 0.5 because fraction need to be represented as decimal value in KidBright.

4. The set of command blocks shown in Figure 4.14 gives the result of 150 as shown in Figure 4.15.

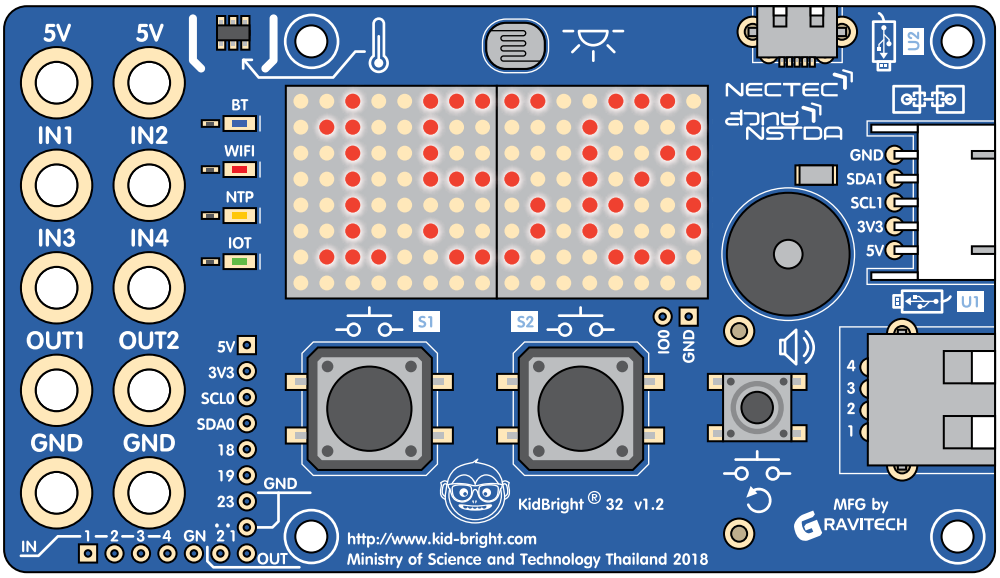


Figure 4.15 Result of the triangle area calculation.

Activity 4.5

Finding an area of a triangle (Add-on)

Students can enhance the presentation of the calculation by incorporating the knowledge from the first few chapters, for example, by creating moving texts to display the calculation step and results, as shown in Figure 4.16.

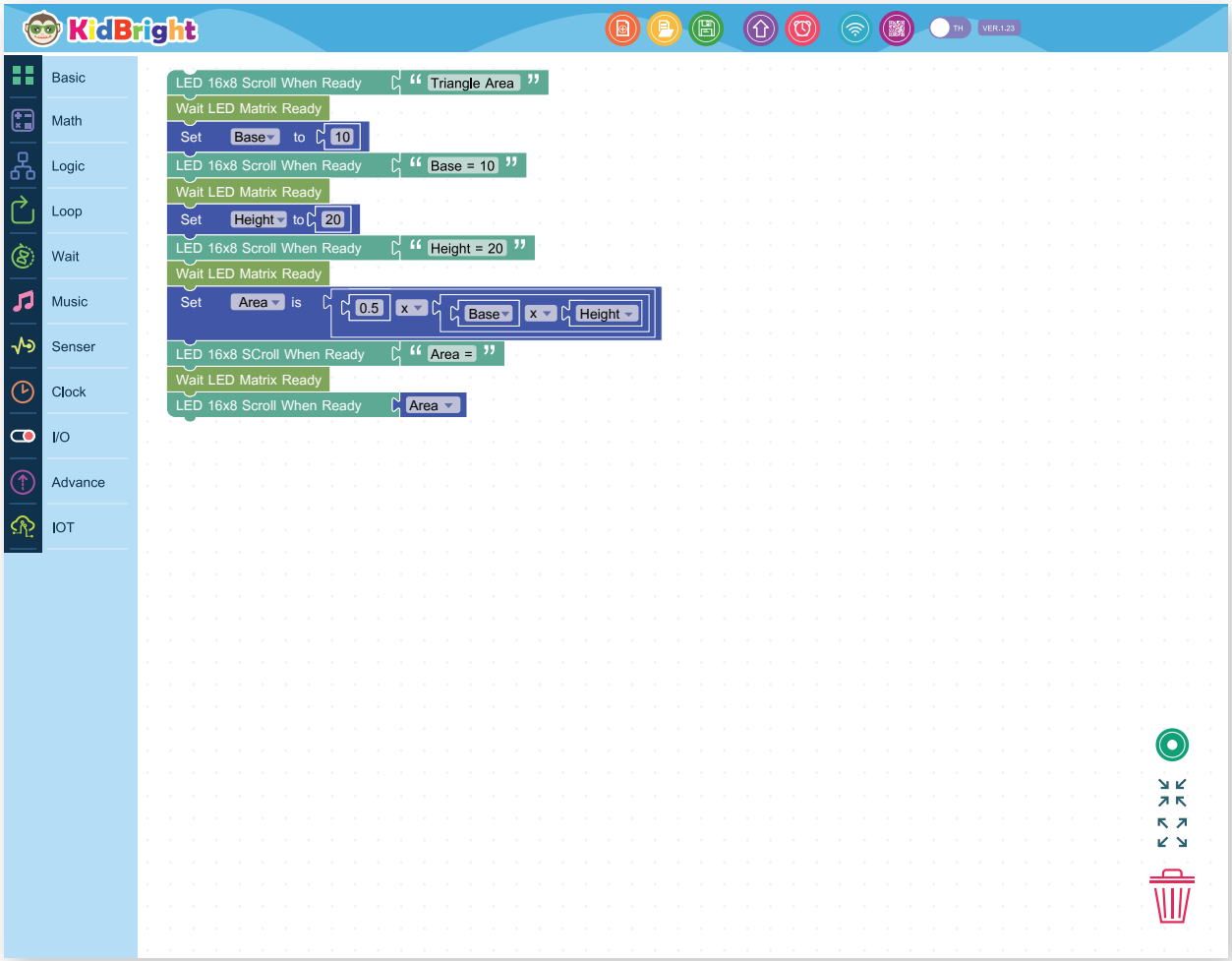


Figure 4.16 Result of the triangle area calculation (Add-on).

Activity 4.6

Finding an area of a rectangle

To find an area of a rectangle as shown in Figure 4.17, we must know the width and the length of rectangle's sides. The calculation to find the area of a rectangle can be done by using following formulae.

Rectangle area = width × length (4.6) or in short as

Rectangle area = W × L (4.7)

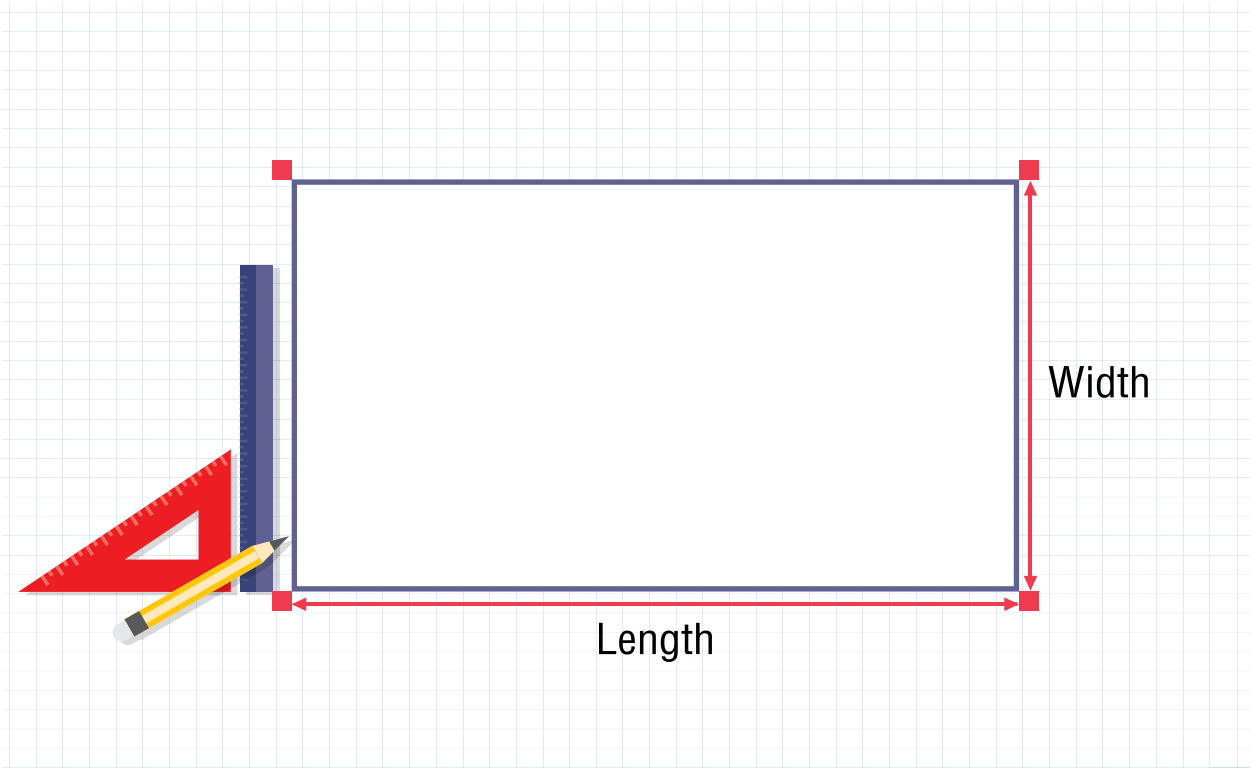


Figure 4.17 Calculating an area of a rectangle.

To calculate the area of a rectangle by using Kidbright, users can use 'Mathematical calculation' block commands in Kidbright IDE as follows:

1. Create 3 different variables named as **Width, Length,** and **Area** by using the 'Variable' block command.
2. Set the values to the **Width** and **Length** variables by assigning the width value to be 10 and the length value to be 30.
3. Based on equation (4.7), create a set of command blocks as shown in Figure 4.18.

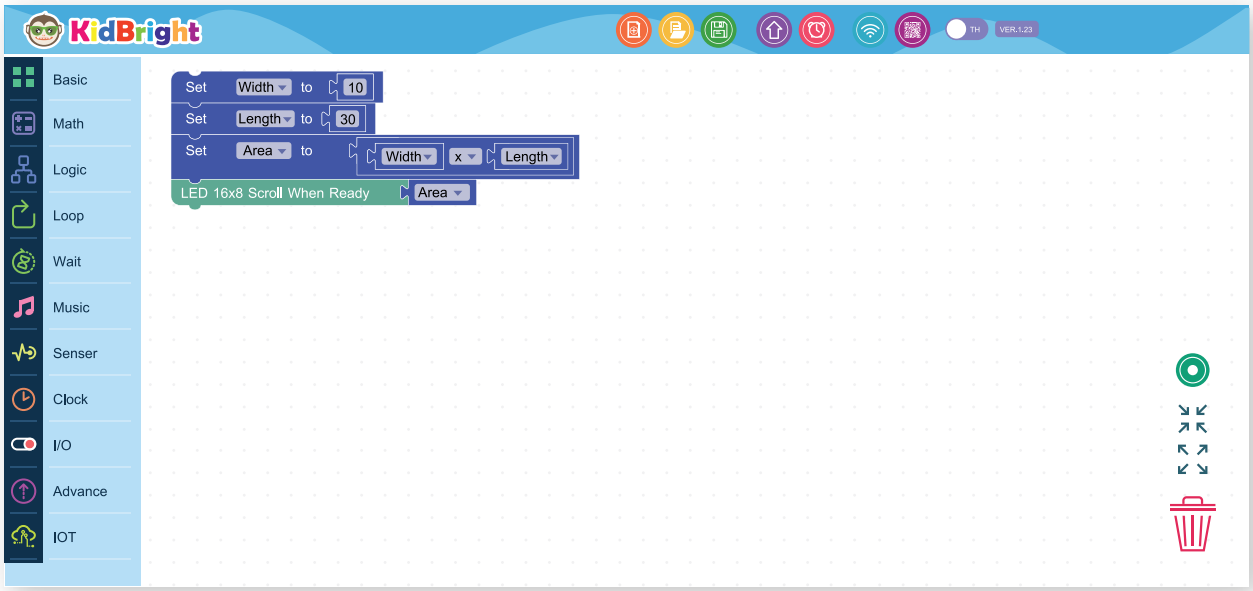


Figure 4.18 Command blocks for finding an area of a rectangle.

4. Based on the above command set shown in Figure 4.18, the result is 300, as shown in Figure 4.19.

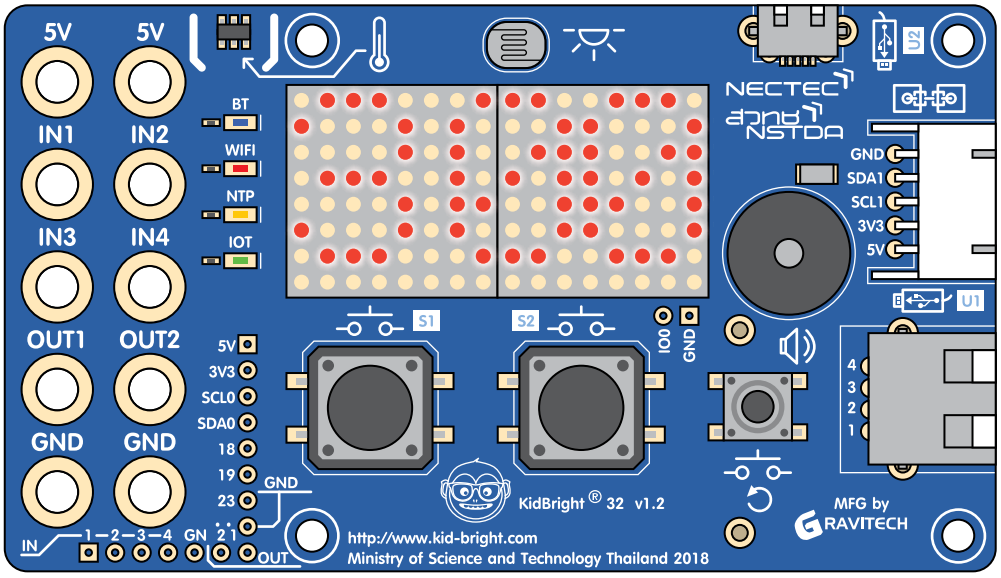


Figure 4.19 Result of a rectangle area calculation.

Similar to Activity 4.5, students can make the result more interesting by applying the moving text function, for example, using 'LED display When Ready block and 'Wait LED matrix ready' block, as shown in Figure 4.20.

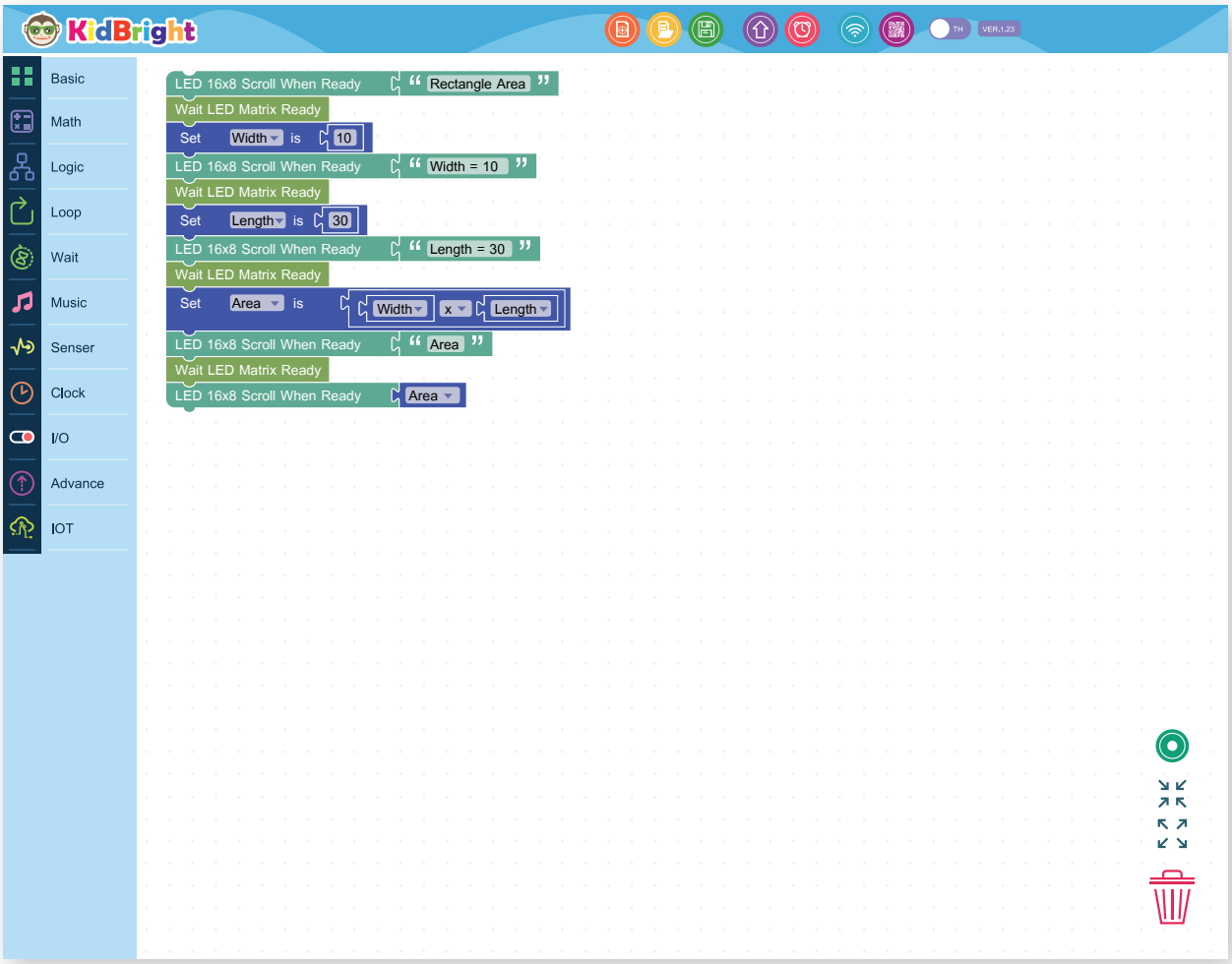
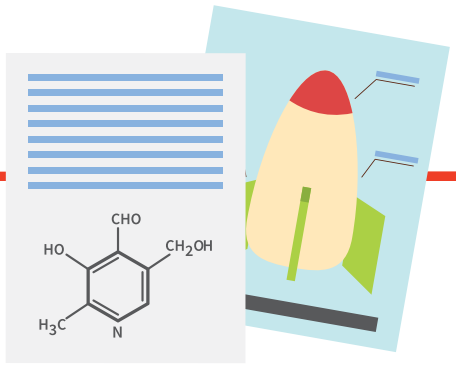


Figure 4.20 Command blocks for finding area of a rectangle (Add-on).

Conclusion

Using 'Mathematical calculation' blocks in Kidbright IDE, we can write command sets for mathematical calculation. The command blocks can be used to create variables, set values of variables, and perform math calculations e.g. addition, subtraction, multiplication, division, and modulo.

Exercise



1. Write a program to calculate the shading area of the square shown in the Figure 4.12.

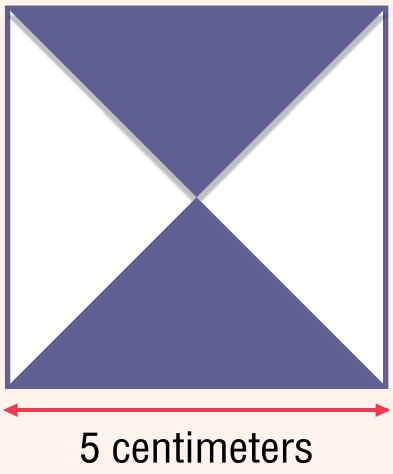


Figure 4.21 A square with the length of 5 centimeters.

2. Write a program to calculate the area of rhombus with the height of 5 centimeters and length of 7 centimeters as shown in Figure 4.22.

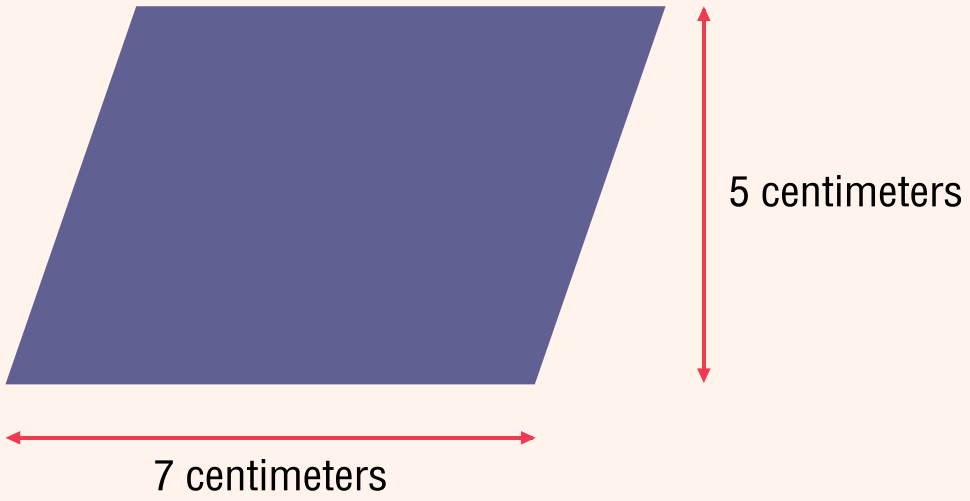


Figure 4.22 A rhombus with the height of 5 centimeters and lenght of 7 centimeters.

Chapter 5
Conditional and Loop Statements

Learning Objectives

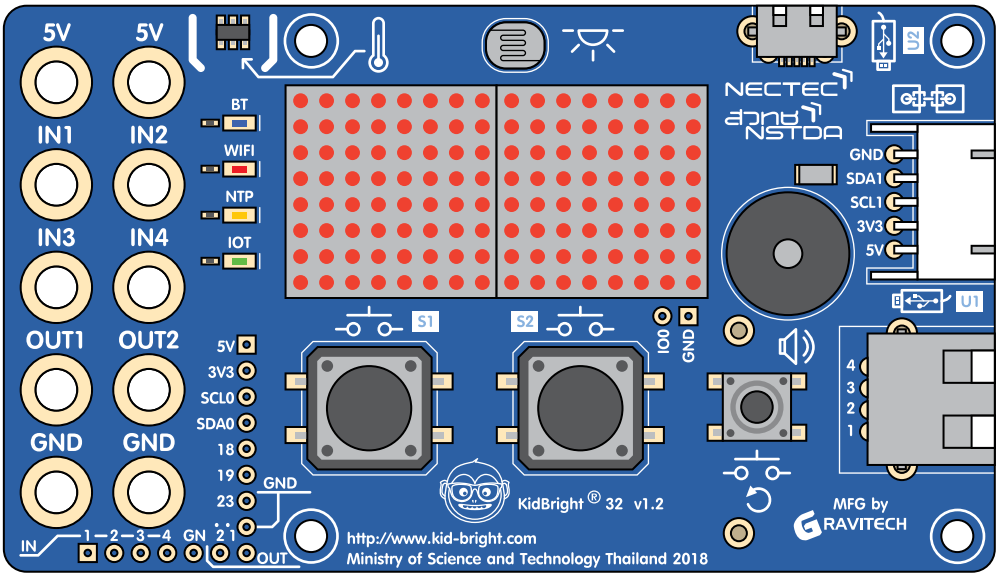
At the end of the learning process of Chapter 5, students are able to:

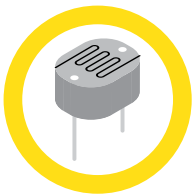
1. Use a temperature sensor, a light intensity sensor, and switches on the KidBright board.
2. Write conditional commands.
3. Write programs with loops.

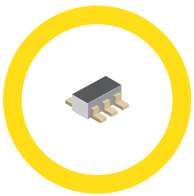
Learning Content

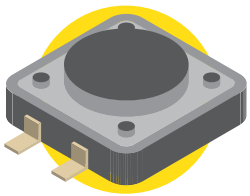
Sensors and their uses

In the embedded system, sensors are devices that are used to detect physical and environmental parameters, such as temperature, light intensity, sound level, and physical contact. Sensors convert these parameters to electrical signals. Furthermore, the signal levels can be used in automatic control systems. In addition, these signals can be recorded for future use. Examples of sensors include light intensity sensors, sound level sensors, magnetic field intensity sensors, temperature sensors, and humidity sensors.









!

The KidBright board has three sensors: a light intensity sensor, a temperature sensor and two manually-controlled switches.

Light intensity sensor

The light sensor on the KidBright board is called a light-dependent resistor (LDR), and is shown in Figure 5.1 (a). The resistance of the LDR decreases when the intensity of the light incident on the LDR increases. We can use the 'Light Level Sensor' block in KidBright IDE for reading the LDR value. Values read from this sensor are represented in the range from 0% (darkest) to 100% (brightest) as shown in Figure 5.1 (b).

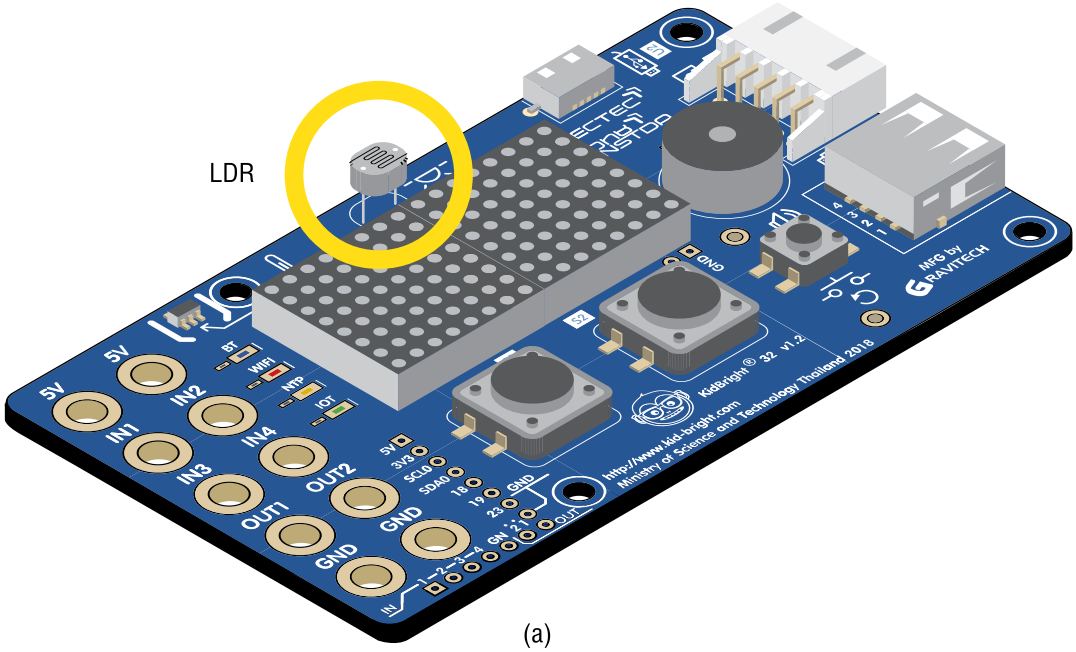
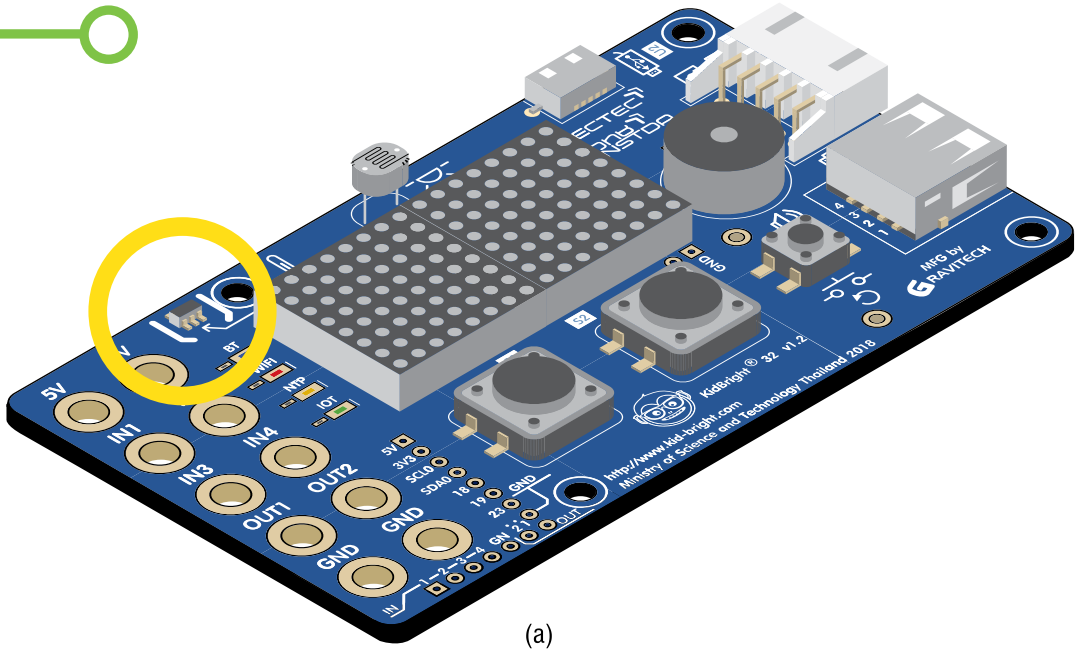


Figure 5.1 (a) Light-dependent resistor (LDR) (b) 'Light Level Sensor' block.

Temperature sensor

On the KidBright board, there is a semiconductor sensor (LM73), as shown in Figure 5.2 (a). This sensor can be used to measure the temperature from 10 to 80 degrees Celsius, with the accuracy of +/- 1.0 degree and the resolution of 0.25 degrees Celsius. The 'Temperature Sensor' block in KidBright IDE can be used to read the value from the temperature sensor as shown in Figure 5.2 (b).



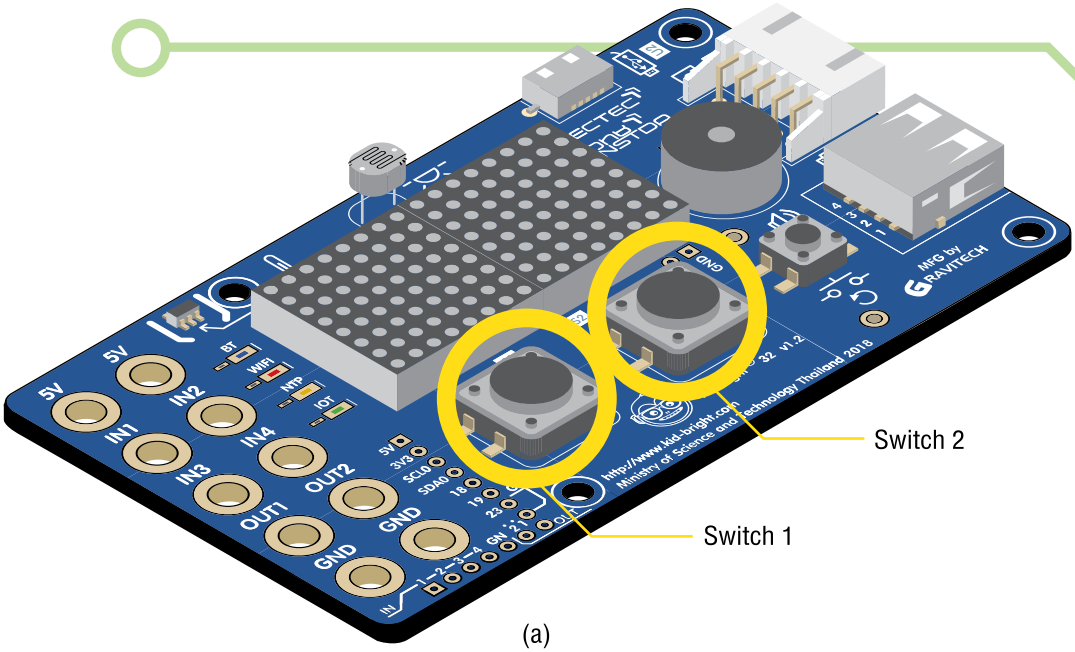
(a)

(b)

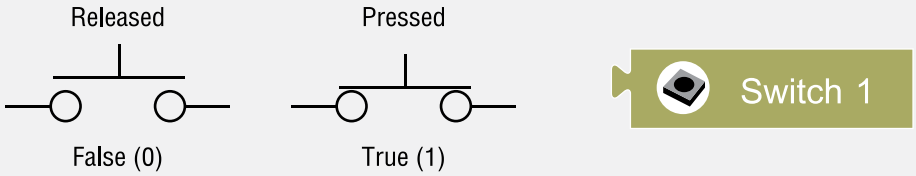
Figure 5.2 (a) Temperature sensor and (b) the 'Temperature Sensor' block.

Switch

There are two manually-controlled switches on the KidBright board, namely Switch 1 and Switch 2, as shown in Figure 5.3 (a). These switches connect the circuit when they are pressed and break the circuit when they are released. The 'Switch' block, as shown in Figure 5.3 (b), is used to read the switch status. When the switch is pressed, the switch value is 1; otherwise, 0.



(a)



(b)

Figure 5.3 (a) Switches and (b) the 'Switch' block.

Conditional statements (or conditional commands)

A conditional statement is used when we want to control results which correspond to the required conditions. For example, if the temperature of our body is greater than 37 degrees Celsius, we might have a fever. We can use KidBright to program a warning message that will be displayed on the screen when the temperature is too high. Thus, the program has to compare a measured temperature with a constant (in this example, it is 37). If the condition is true (i.e., the measured temperature is higher than 37 degrees Celsius), the KidBright board will display an alarm message.

The KidBright IDE provides three conditional blocks as follows:

1. 'if' block
2. 'if-else' block
3. 'if-elseif-else' block (or a nested 'if-else' block)

'if' block

The 'if' block checks a condition that keeps behind the word 'if'. If the condition is true, all commands under the 'do' will be executed; otherwise, they will be skipped.

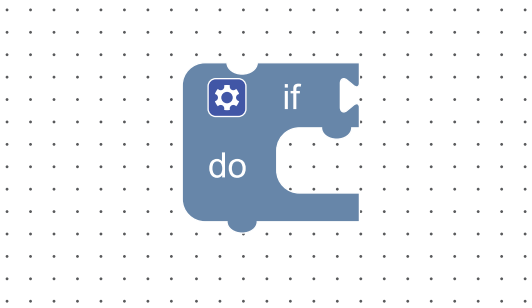


Figure 5.4 'if' block.

Activity

Activity 5.1

A program that reads the light intensity level and checks a condition by using the 'if' block as shown in Figure 5.5.

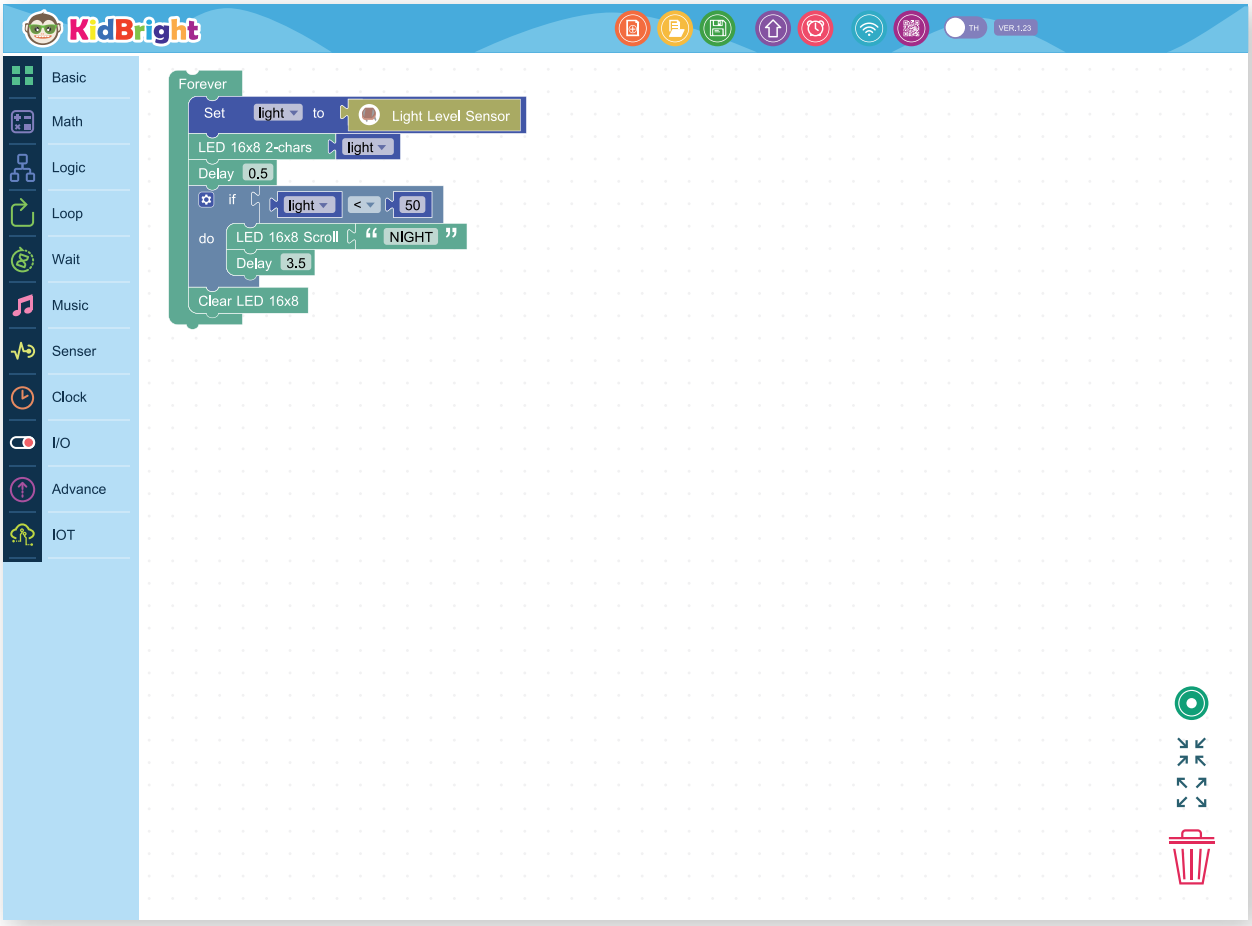


Figure 5.5 Program for reading the light intensity and checking a condition by using the 'if' block.

Description of program in Figure 5.5.

1. Set a forever loop.
2. Set a variable, named as **light**, for storing the light intensity level.
3. Display the light value on the 'LED 16x8 2-chars' block.
4. Delay for 0.5 second.
5. Check a condition.
   5.1 If the condition is true (i.e., the light value is less than 50).
      5.1.1 Display a text "NIGHT" on the screen.
      5.1.2 Delay for 3.5 seconds.
6. Clear the screen LED 16x8.
7. Go to step 2.

‘if-else’ block

In checking a condition of ‘if’, if the condition is true, all commands in a block ‘do’ are executed. But if the condition is false, all commands in a block ‘else’ are executed. The ‘if-else’ block is shown in Figure 5.6.

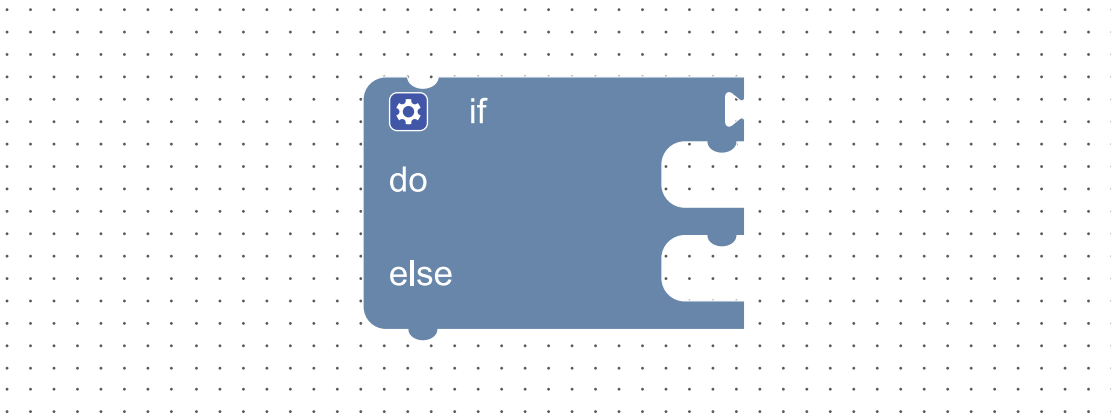


Figure 5.6 ‘if-else’ block.

Activity 5.2

A program that reads the light intensity level and checks a condition by using the 'if-else' block as shown in Figure 5.7.

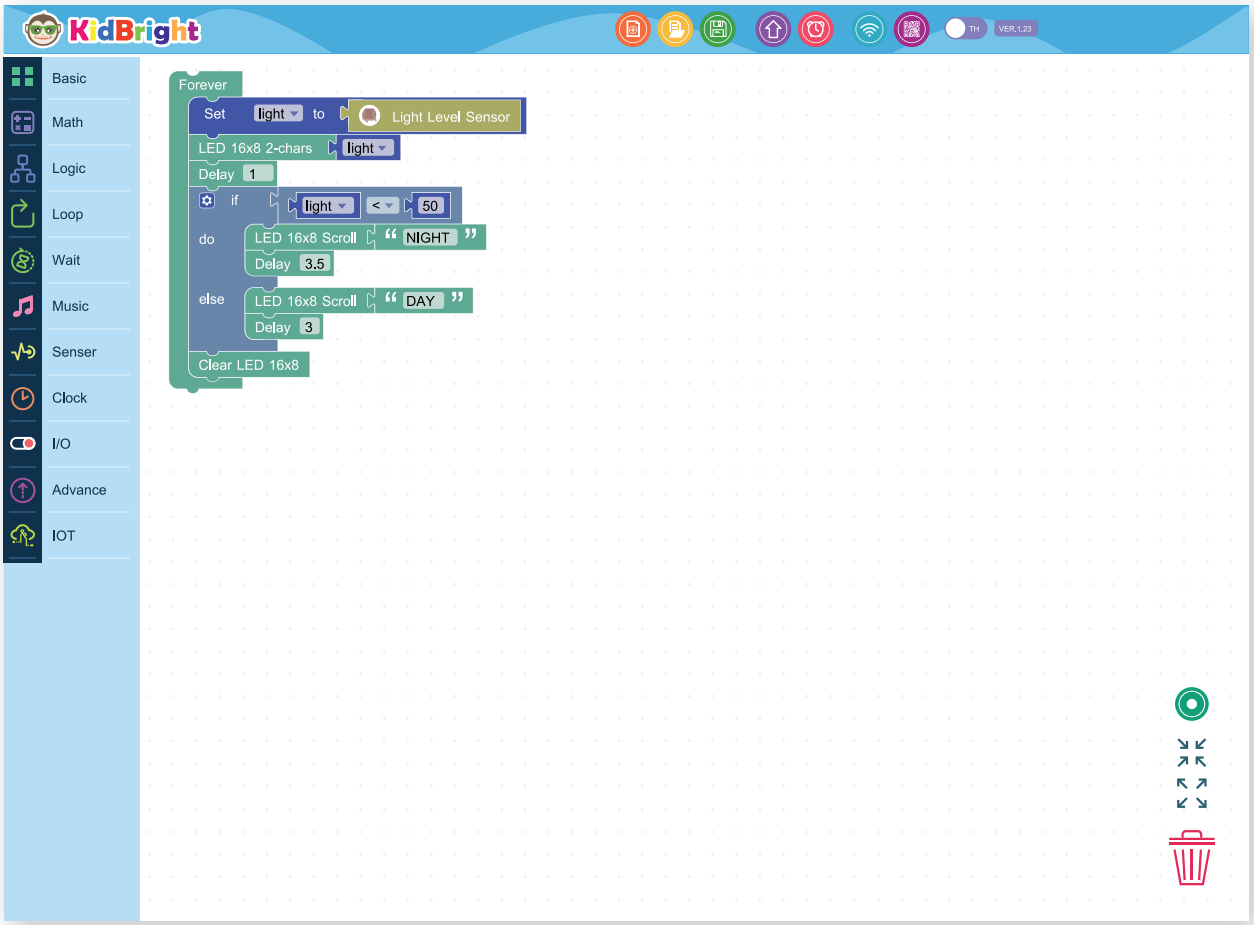


Figure 5.7 Program for reading the light intensity and checking a condition by using the 'if-else' block.

Description of program in Figure 5.7.

1. Set a forever loop.
2. Set a variable, named as **light**, for storing the light intensity level.
3. Display the light value on the 'LED 16x8 2-chars' block.
4. Delay for 1 second.
5. Check a condition.
   - 5.1 If the condition is true (i.e., the light value is less than 50).
     - 5.1.1 Display a text "NIGHT".
     - 5.1.2 Delay for 3.5 seconds.
   - 5.2 If the condition is false (i.e., the light value is higher than or equal to 50).
     - 5.2.1 Display a text "DAY".
     - 5.2.2 Delay for 3 seconds.
6. Clear the screen.
7. Go to step 2.

'if-elseif-else' block

This block is used when there are more than 2 choices (i.e., when you want to check more than one condition). You can add additional conditions after the 'else if' block. The 'else if' block can be added by clicking the setting button and dragging the 'else if' block (or the 'else' block on the left-hand side) to drop under the 'if' block on the right-hand side. Furthermore, the 'else if' blocks can add as many as the number of required conditions. Note that 'if-elseif-else' block is shown in Figure 5.8.

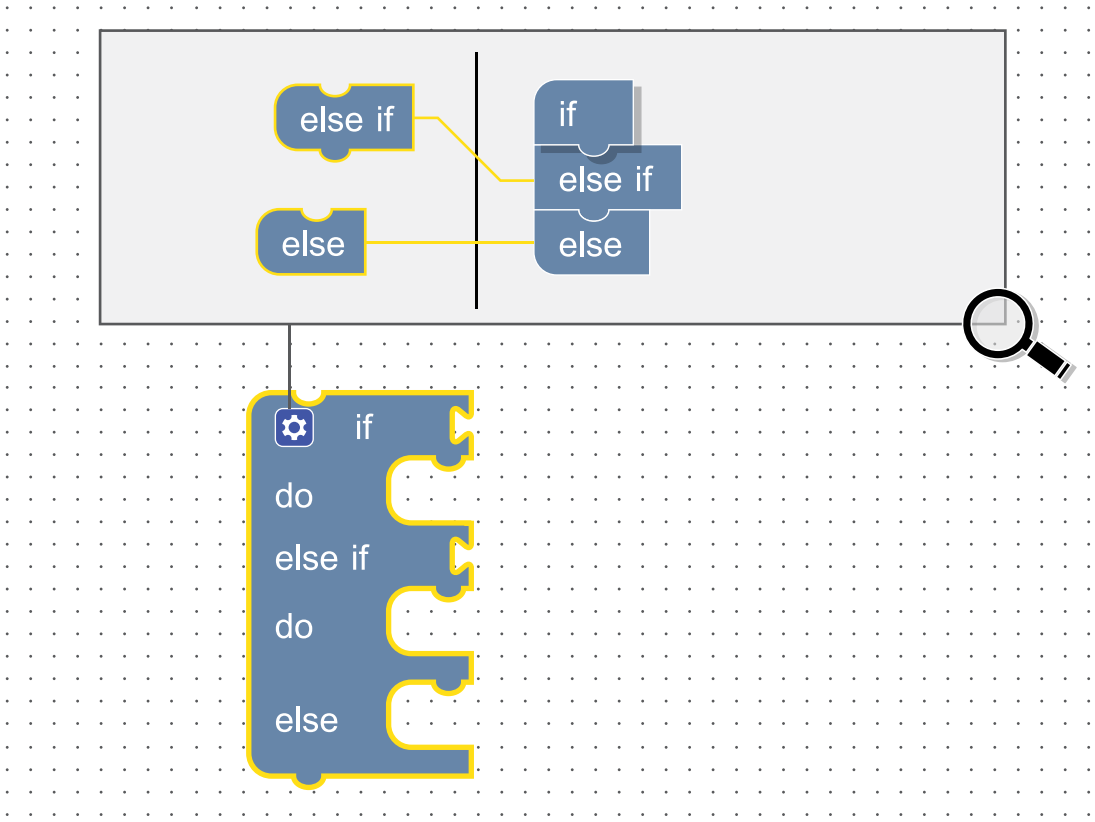


Figure 5.8 'if-elseif-else' block.

Activity 5.3

A program that reads the light intensity level and checks a condition by using the 'if-elseif-else' block as shown in Figure 5.9.

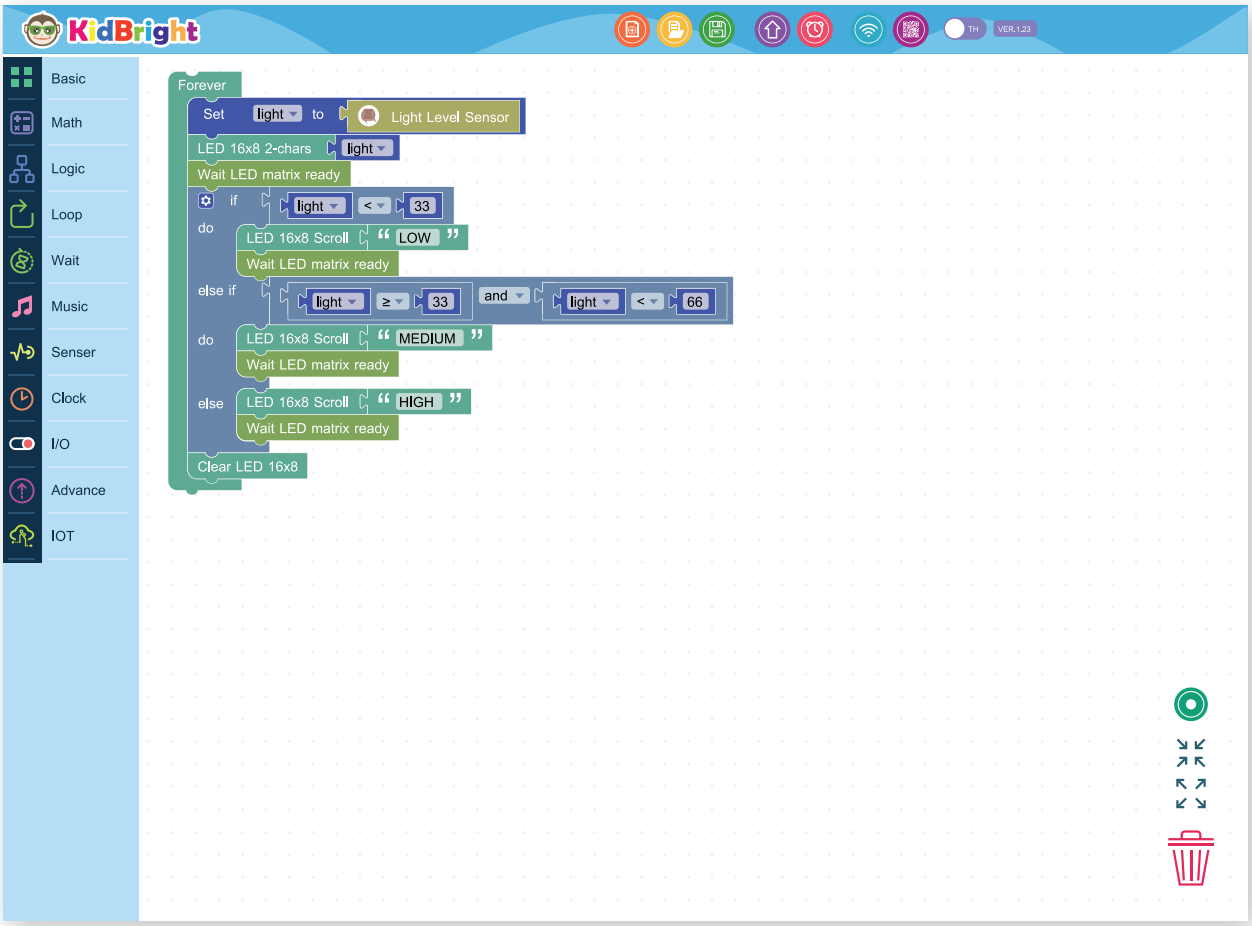


Figure 5.9 Program for reading the light intensity and checking a condition by using the 'if-elseif-else' block.

Description of program in Figure 5.9.

1. Set a forever loop.
2. Set a variable, named as **light**, for storing the light intensity level.
3. Display the light value on the 'LED 16x8 2-chars' block.
4. Delay for 1 second.
5. Check a condition.
   - 5.1 If the light value is less than 33.
     - 5.1.1 Display the text "LOW".
     - 5.1.2 Wait until the last character is on the screen.
   - 5.2 If the light value is higher than or equal to 33 but less than 66.
     - 5.2.1 Display the text "MEDIUM".
     - 5.2.2 Wait until the last character is on the screen.
   - 5.3 If the light value is greater than or equal to 66.
     - 5.3.1 Display the text "HIGH".
     - 5.3.2 Wait until the last character is on the screen.
6. Clear the screen.
7. Go to step 2.

Loop statements

Commands under a loop statement (a repeat statement or a loop block) can be executed more than one time depending on the loop condition. The KidBright IDE provides two conditional loop blocks as follows:

1. The 'Repeat while' block as shown in Figure 5.10 allows all commands under it to perform repeatedly as long as a condition that follows the 'Repeat while' block is true.

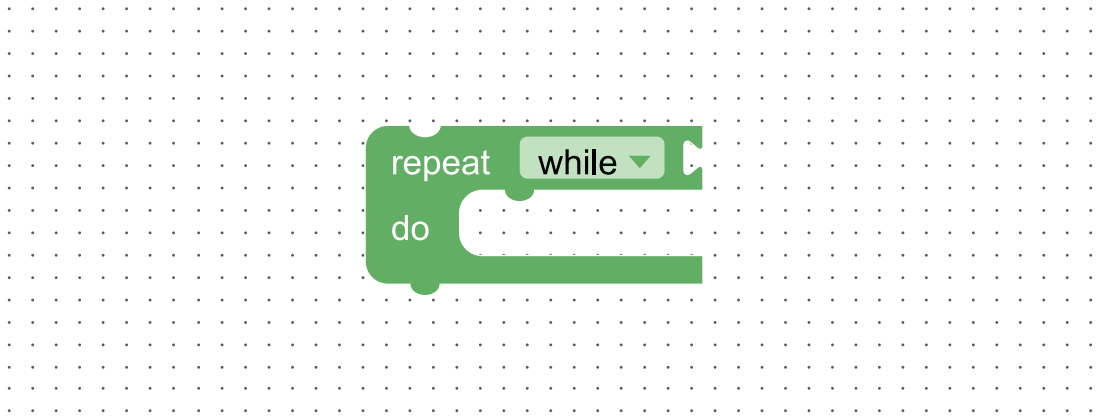


Figure 5.10 'Repeat while' block.

2. The 'Repeat until' block as shown in Figure 5.11 allows all commands under it to perform repeatedly as long as a condition that follows the 'Repeat while' block is false.

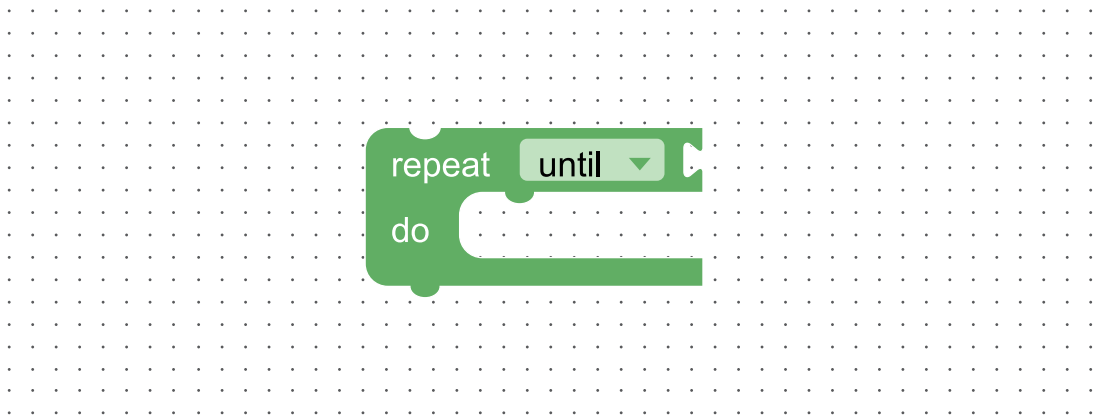


Figure 5.11 'Repeat until' block.

However, you can force the program to exit the loop by using the 'Break' block. 

Activity 5.4

A program that displays numbers from 0 to 7 on the screen when Switch 1 is pressed.

Write the program without using the 'Repeat while' block as shown in Figure 5.12.

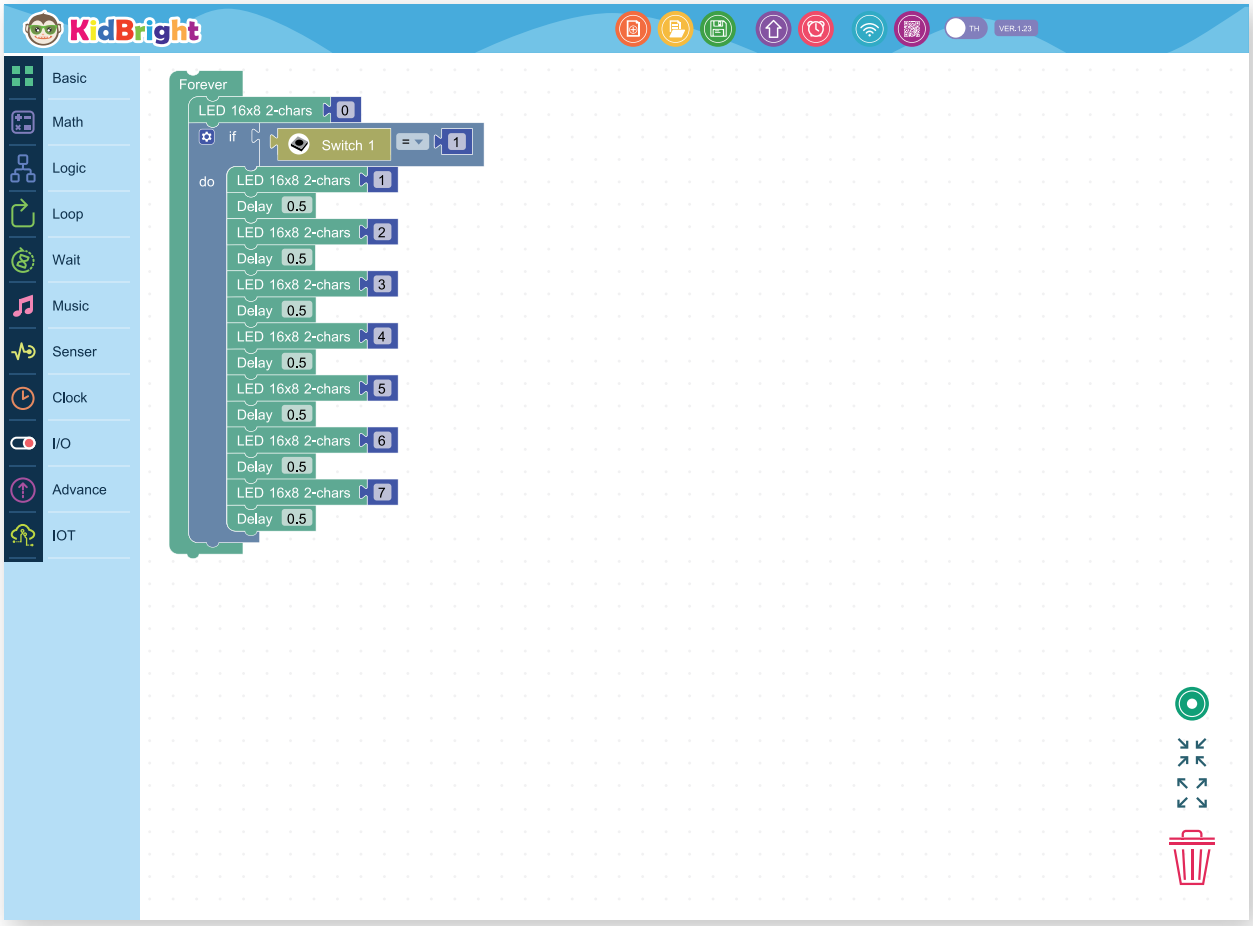


Figure 5.12 Program for displaying the numbers from 0 to 7 on the screen when Switch 1 is pressed.

Description of program in Figure 5.12.

1. Set a forever loop.
2. Display “0” on the ‘LED 16x8 2-chars’ block.
3. Check a condition ‘if Switch 1 is pressed.’ If the condition is true, do the following command.
   - 3.1 Display “1” on the ‘LED 16x8 2-chars’ block.
   - 3.2 Delay for 0.5 second.
   - 3.3 Display “2” on the ‘LED 16x8 2-chars’ block.
   - 3.4 Delay for 0.5 second.
   - 3.5 Display “3” on the ‘LED 16x8 2-chars’ block.
   - 3.6 Delay for 0.5 second.
   - 3.7 Display “4” on the ‘LED 16x8 2-chars’ block.
   - 3.8 Delay for 0.5 second.
   - 3.9 Display “5” on the ‘LED 16x8 2-chars’ block.
   - 3.10 Delay for 0.5 second.
   - 3.11 Display “6” on the ‘LED 16x8 2-chars’ block.
   - 3.12 Delay for 0.5 second.
   - 3.13 Display “7” on the ‘LED 16x8 2-chars’ block.
   - 3.14 Delay for 0.5 second.
4. Go to step 2.

Write the program by using the 'Repeat while' block.

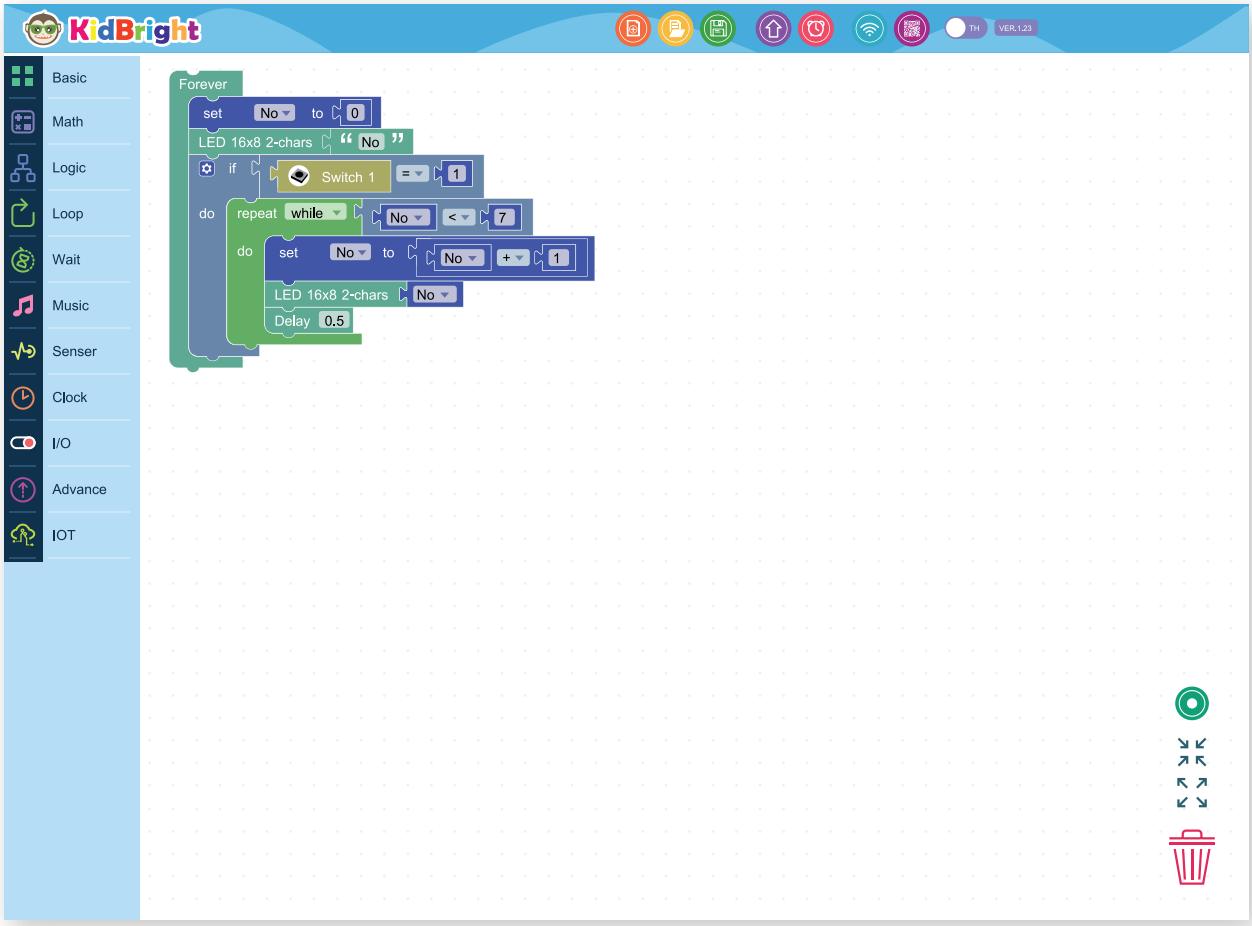


Figure 5.13 Commands when the 'Repeat while' block is used.

Description of program in Figure 5.13.

1. Set a forever loop.
2. Create a variable, called **No**, and set it to 0.
3. Display the **No** value on the 'LED 16x8 2-chars' block.
4. Check a condition 'if Switch 1 is pressed.' If the condition is true (i.e., the switch is pressed), do the following commands:
   4.1. If the condition is true, increase the value of **No** by 1.
   4.2. Display the **No** value on the 'LED 16x8 2-chars' block.
   4.3. Delay for 0.5 second.
   Then, the commands under this loop are repeatedly executed until the condition is false (i.e., until the **No** value is higher than or equal to 7).
5. Go to step 2.

Write a program by using the 'Repeat while' block that exits the loop if Switch 2 is pressed.

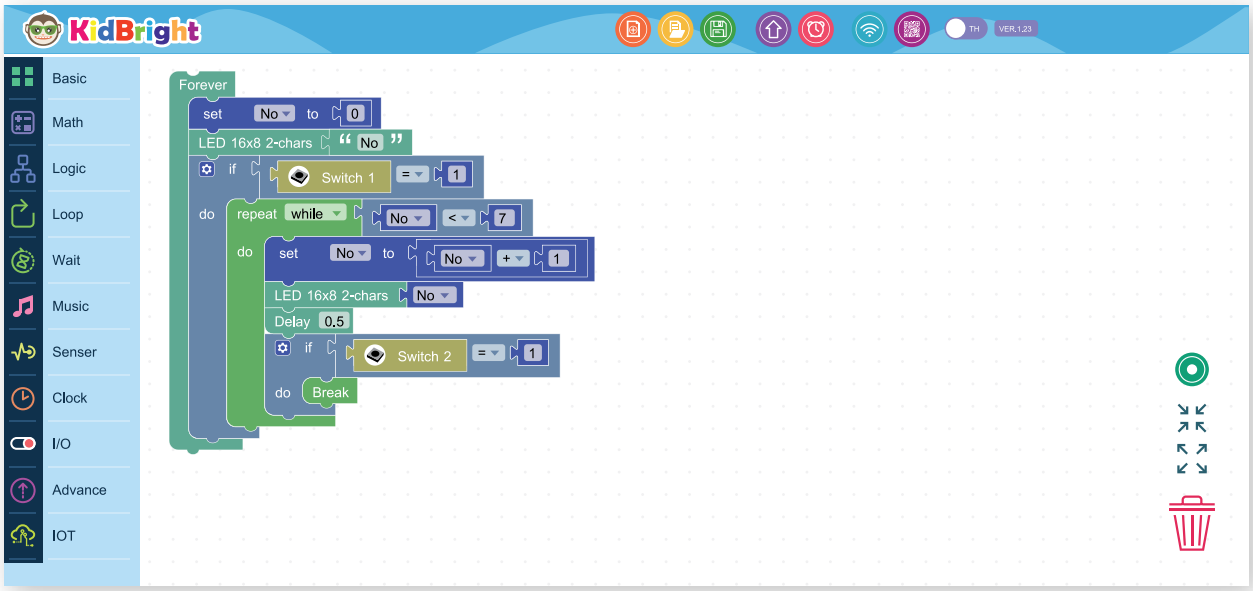


Figure 5.14 Commands used in the program that exits the loop if Switch 2 is pressed.

Description of program in Figure 5.14.

1. Set a forever loop.
2. Create a variable, called **No**, and set it to 0.
3. Display the **No** value on the 'LED 16x8 2-chars' block.
4. Check a condition 'if Switch 1 is pressed.'
   If the condition is true (i.e., Switch 1 is pressed), do the following commands.
   - 4.1 Check a condition 'if the No value is less than 7'
     - 4.1.1 If the value of **No** is less than 7, increase its value by 1.
     - 4.1.2 Display the **No** value on the 'LED 16x8 2-chars' block.
     - 4.1.3 Delay for 0.5 second.
     - 4.1.4 Check a condition 'if Switch 2 is pressed'
       - 4.1.4.1 If the condition is true, do the 'Break' block (i.e., stop checking the value of **No**) and go to step 4.2.
       - 4.1.4.2 If the condition is false, continue repeatedly checking the value of **No** until it is greater than or equal to 7 and then go to step 4.2.
   - 4.2 Go to step 2.

Activity 5.5

A program for measuring the temperature and controlling an electric fan is shown in Figure 5.15.

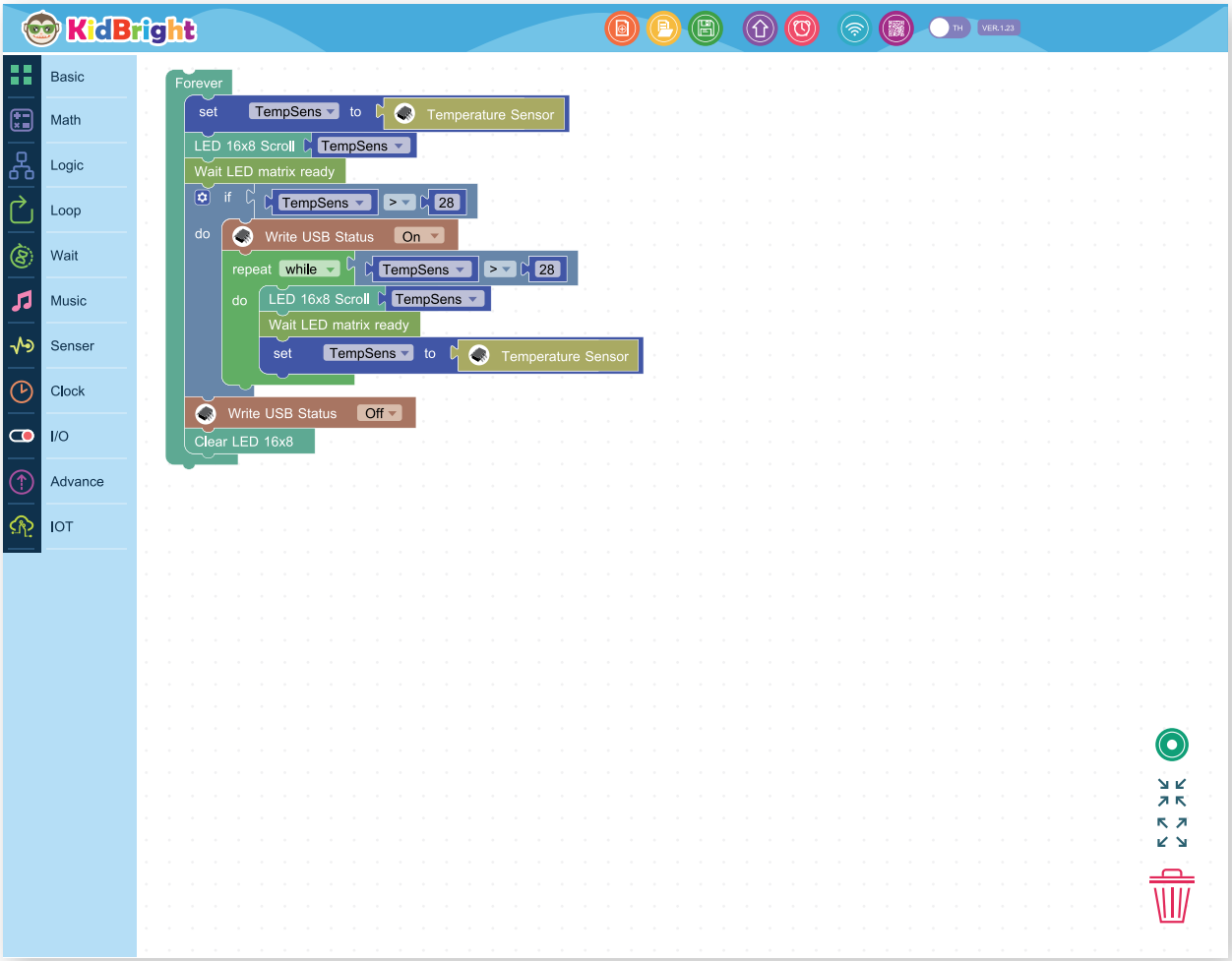


Figure 5.15 Commands used in the program for measuring the temperature and turning on an electric fan.

Description of program in Figure 5.15.

1. Set a forever loop.
2. Create a variable, named as **TempSens**, for storing the temperature.
3. Display the **TempSens** value on the 'LED 16x8 Scroll' block.
4. Wait until the last character is displayed.
5. Check a condition. If the condition is true (i.e., the measured temperature is greater than 28).
   - 5.1 Turn the USB port on so that the fan is turned on.
   - 5.2 Repeatedly do commands under the 'Repeat while' block as long as the condition is true (i.e., the temperature is higher than 28). And exit the loop when the condition is false. The commands under the 'Repeat while' block are as follows.
     - 5.2.1 Display the **TempSens** value on the screen.
     - 5.2.2 Wait until the last character is on the screen.
     - 5.2.3 Read the temperature and assign it to the variable **TempSens**.
6. Turn the USB port off so that the fan is turned off.
7. Clear the screen.
8. Go to step 2.

Activity 5.6

A program that measures the temperature and turns on an electric fan with a condition that if Switch 1 is pressed, the fan will be turned off.

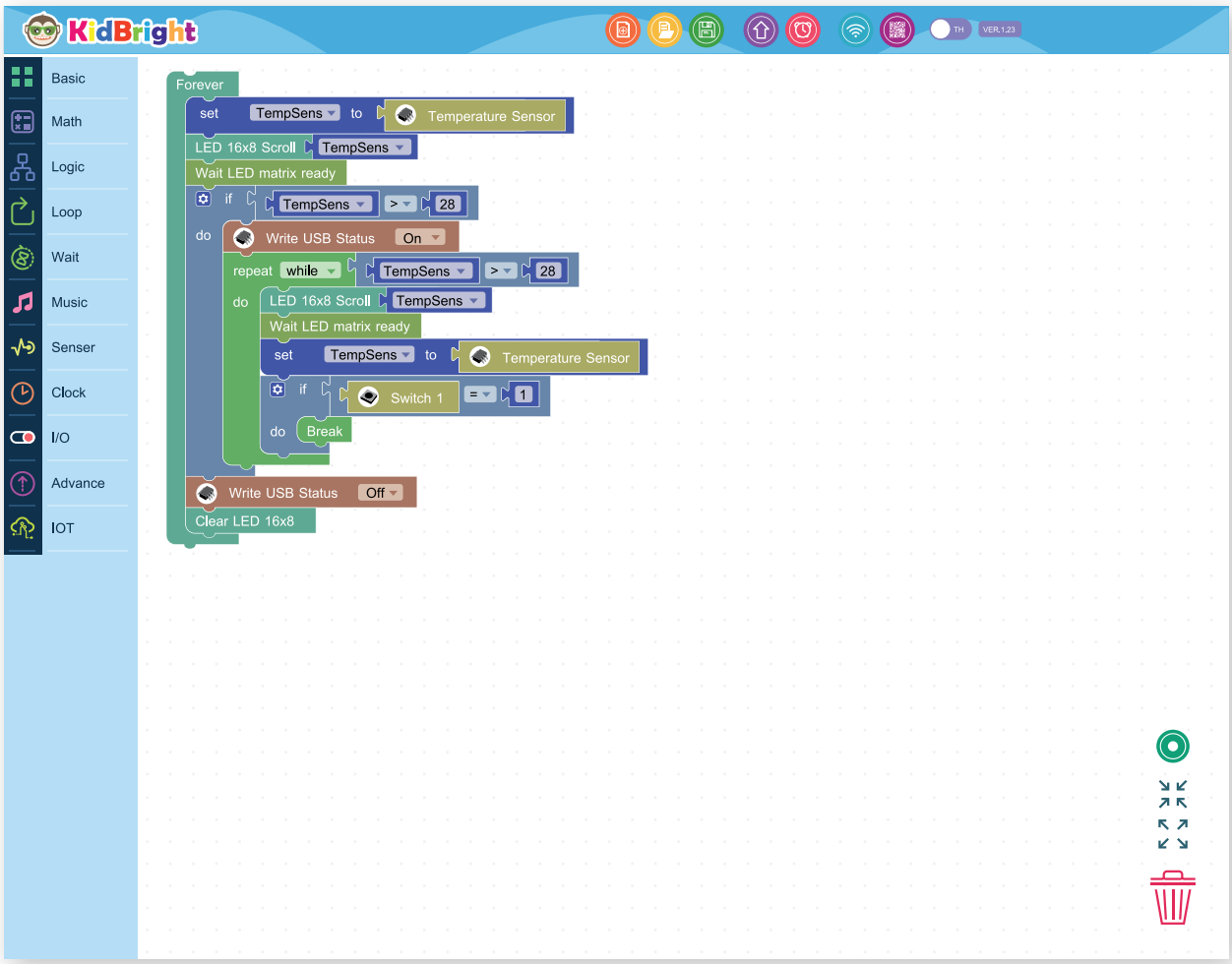


Figure 5.16 Commands used in the program that measures the temperature and turns on an electric fan with the condition that if Switch 1 is pressed, the fan will be turned off.

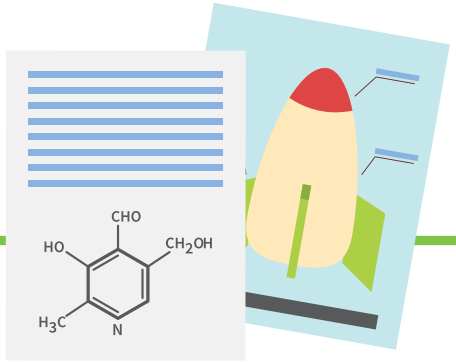
Description of program in Figure 5.16.

This program is similar to the program used in Activity 5.5. The only difference is that there is a conditional check for the status of Switch 1. That is, if Switch 1 is pressed, the fan will be turned off by using the 'Break' block.

Summary

Students learn how to use the temperature sensor, the light intensity sensor, and the switches in this chapter. Also, students are expected to understand how to use the conditional blocks and the loop blocks.

Exercise



Based on the program used in Activity 5.4, write another program for the same task by using the 'Repeat until' block.

Write a program that displays the text “Hot” on the screen when the temperature is higher than 30 degrees Celsius and show a smiling face when the temperature is less than or equal to 30 degrees Celsius.

Chapter 6
Enter the World of Music

Learning Objectives

At the end of the learning process of Chapter 6, students are able to:

1. Use Music blocks to enable a buzzer for playing musical notes.
2. Read musical notes on a typical five-line staff and transform the notes to programming codes by using Music blocks.
3. Write a repeatable program under the 'Repeat while' condition to repeat any part of a song.

Learning Content

Learn musical notes

Musical notes represent the values of time signature. Notes represent the duration or length of sound (Rhythm) and show the high or low level of sound (Pitch).

Musical note values and rhythms

- Music note symbols

The symbol of each music note has different rhythmic values (number of beats), as shown in Table 6.1

Note symbol	American name	Rhythm (4/4 time signature)
	Whole	4
	Half	2
	Quarter	1
	Eighth	1/2
	Sixteenth	1/4

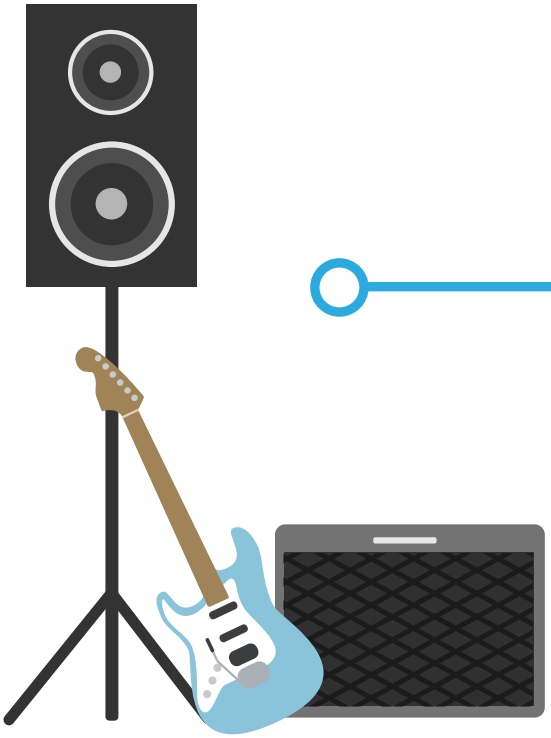
Table 6.1 Music note symbols, names, and rhythms.

- Rest symbols

Rest symbols represent silent intervals according to the rest rhythms. Rest symbols are shown in Table 6.2.

Rest symbol	American name	Rhythm (4/4 time signature)
𝄻	Whole rest	4
𝄼	Half rest	2
𝄽	Quarter rest	1
𝄾	Eighth rest	1/2
𝄿	Sixteenth rest	1/4

Table 6.2 Rest symbols.



Musical note sounds

Musical note sounds can be described by tonal scales of pitch in the sequence Do, Re, Mi, Fa, Sol, La, Ti, which are denoted in English letter notation as shown in Table 6.3.

Note sound	Tonal scale of note sound in English alphabets
Do	C
Re	D
Mi	E
Fa	F
Sol	G
La	A
Ti	B

Table 6.3 Tonal scale of note sounds in English alphabets.

Five-line staff

A staff is made up of a group of five evenly-spaced horizontal lines. Notes are written on the staff in musical notaiton.

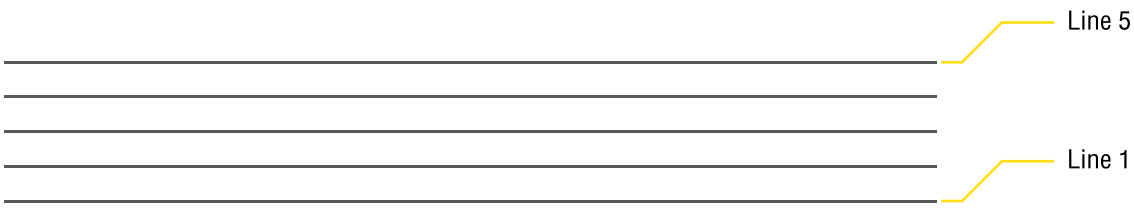


Figure 6.1 Typical five-line staff.

A staff with G Clef is the most common. G Clef is the musical note placed on the second line on the staff, which is the Sol sound tone.

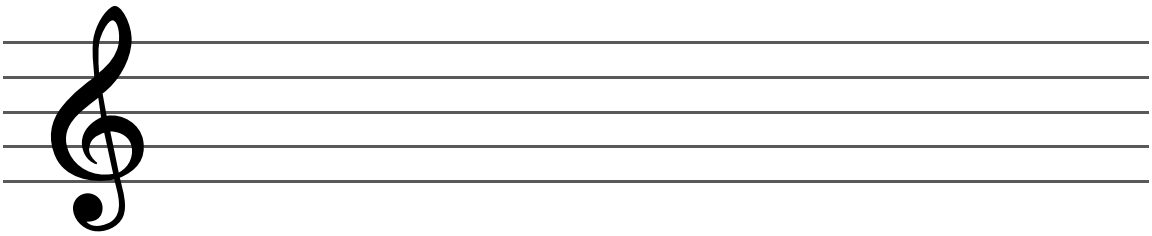


Figure 6.2 Five-line staff with G Clef.

Pitch levels of music notes on a staff with G clef are shown in Figure 6.3.

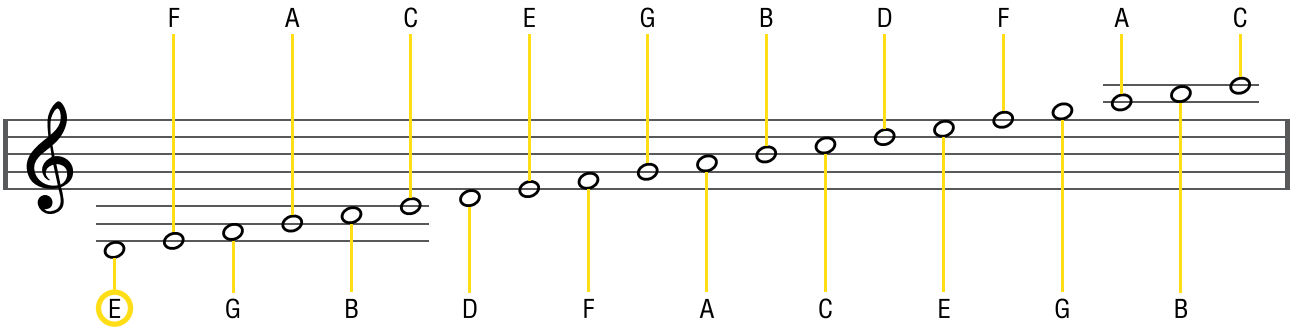


Figure 6.3 Sound levels on a staff with G Clef.

Reading musical notes on a staff

The vertical position of musical notes written on a staff represent their sound levels, whereas note symbols represent the rhythms of the notes. Therefore, two types of information are received when we read a musical note on staff, 1) the pitch of the note and 2) the length of the sound.

An example of reading musical notes on a staff is shown in Figure 6.4.

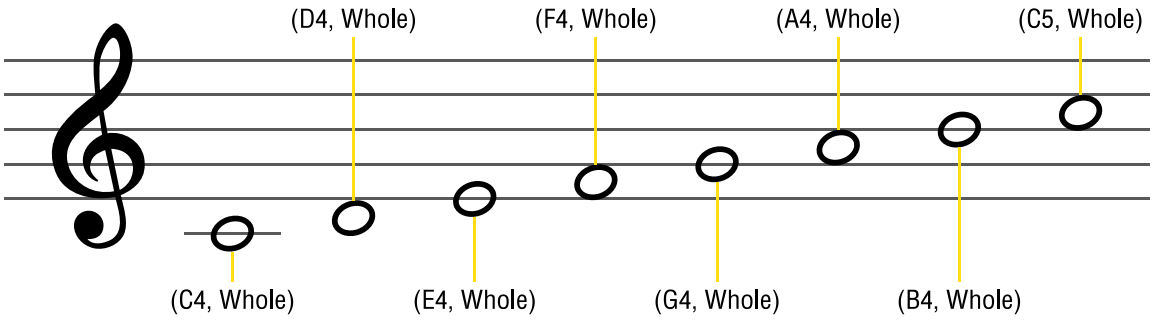


Figure 6.4 Musical notes on a staff.

***Remark** C4 means C note at the fourth octave and C5 means C note at the fifth octave.

Music Blocks in KidBright IDE

The Music tap consists of five command blocks as shown in Figure 6.5. These command blocks enable the buzzer on the KidBright board to produce a sound.

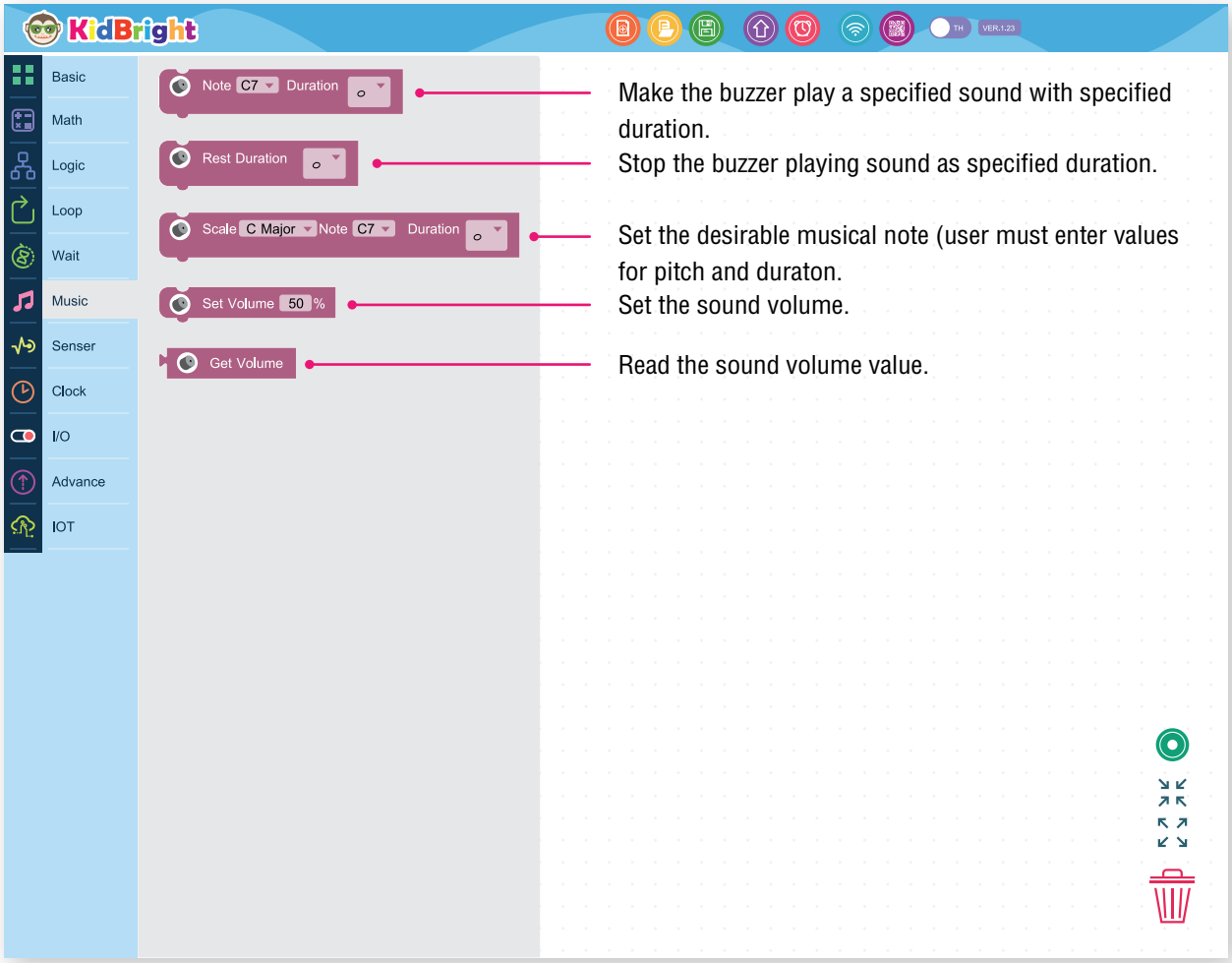


Figure 6.5 Commands blocks in Music tap.

Frequently used blocks

- ‘Sounding’ block

‘Sounding’ block is the block used to play a note sound. To use this block, a musical note pitch needs to be selected, for example, C, D. E, F, G, A or B and the duration of the note sound also has to be specified from Whole, Half, Quarter, Eighth, or Sixteenth.

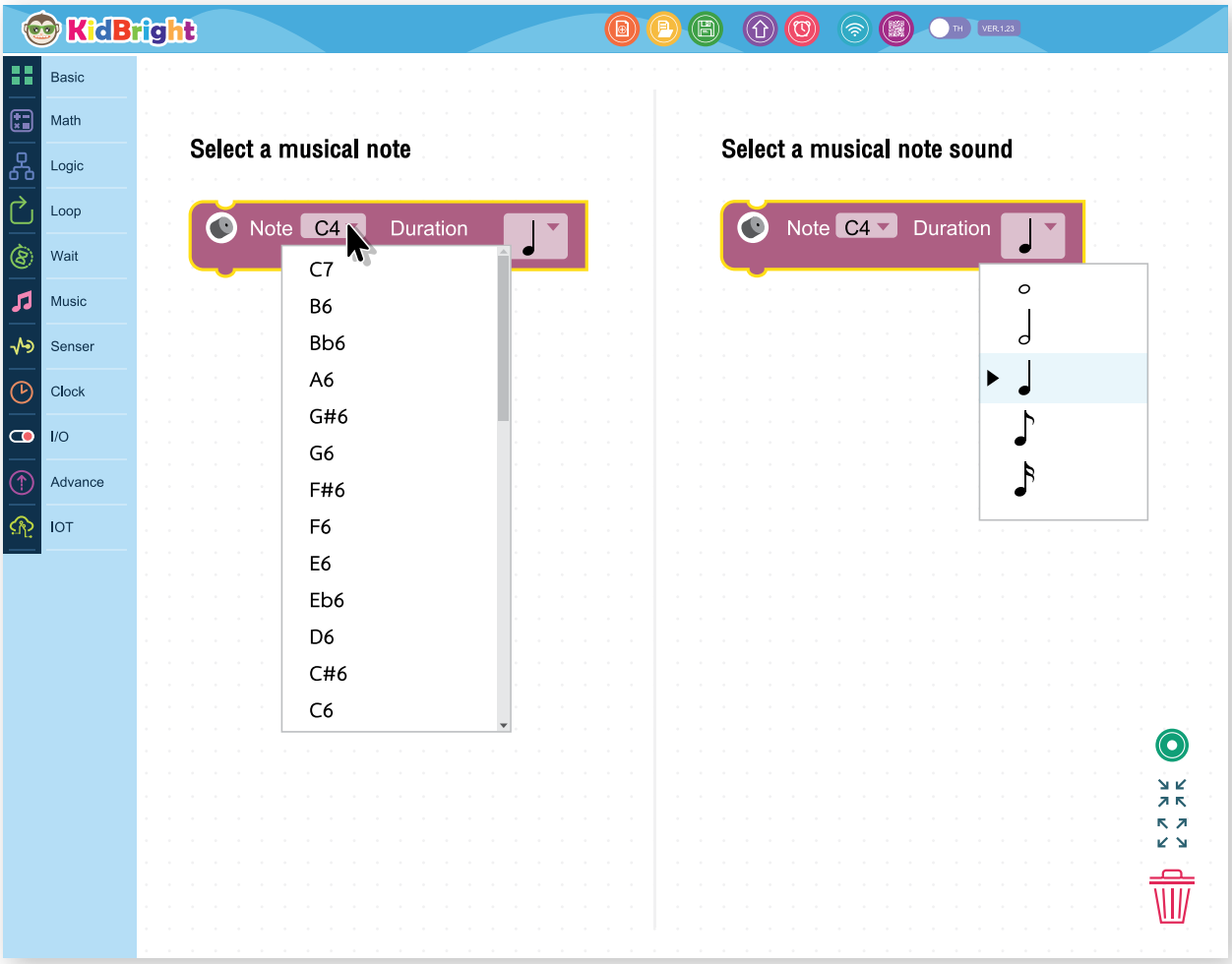


Figure 6.6 How to set up ‘Sounding’ block by selecting a musical note and specifying a duration of the note to be played.

- ‘Resting’ block

‘Resting’ block is the block used for an interval of silence. To use this block, the duration of silence has to be specified from Whole, Half, Quarter, Eighth, or Sixteenth.

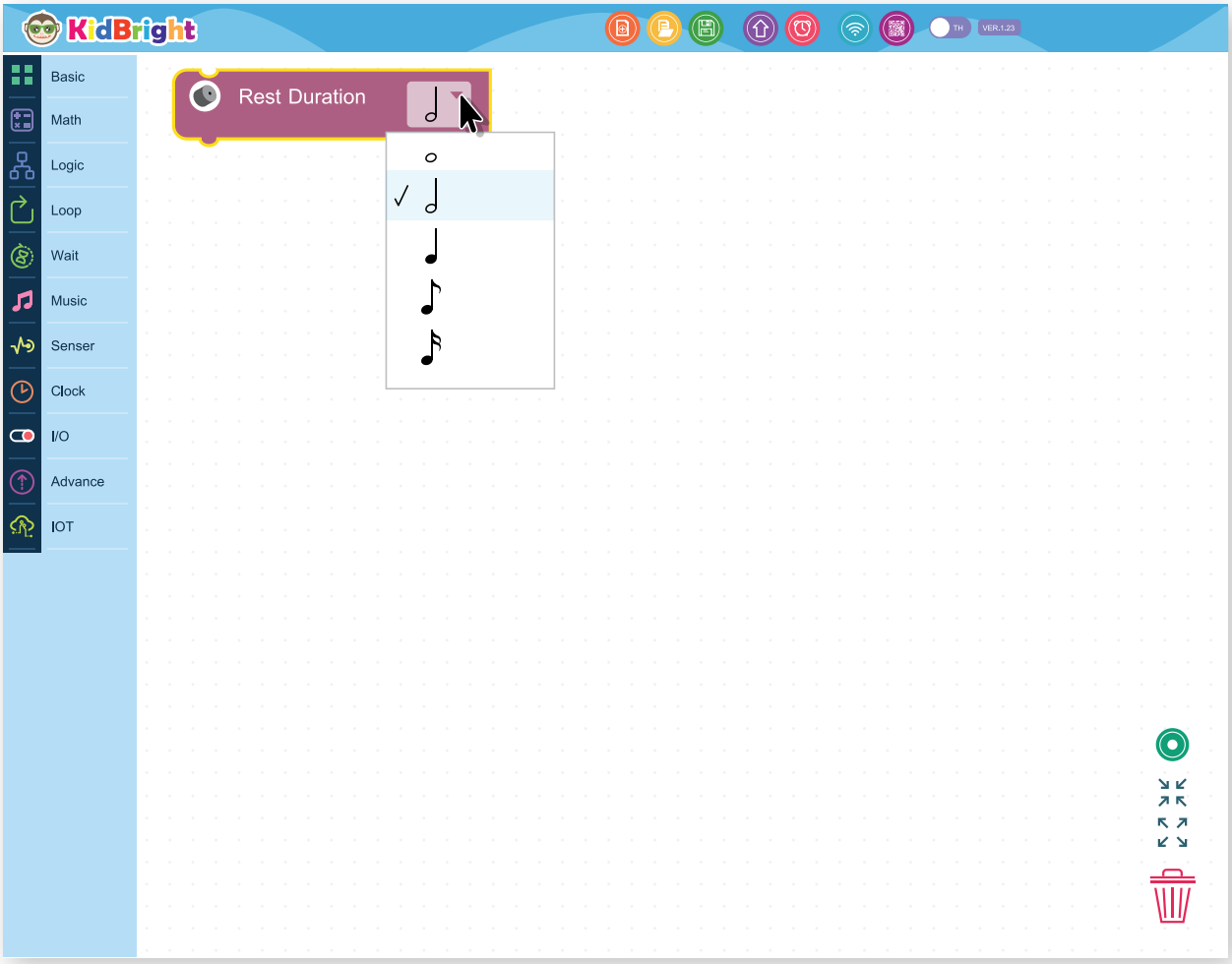


Figure 6.7 How to set up a ‘Resting’ block by specifying the duration of silent.

- ‘Volume setting’ block

‘Volume setting’ block is the block that is used to set a level of sound volume, or how loud the sound is. This is done by specifying a number in percentage. The default is 50%.

Figure 6.8 ‘Volume setting’ block.

Transforming musical notes on a five-line staff to block coding commands

To transform musical notes on a five-line staff to block coding commands, the user needs to read the notes on the staff, assign note pitch values and specify the length of each note. For any rest periods in the music, the rest duration has to be specified in the resting block.

As shown in Figure 6.9, there are three notes on the staff. Therefore, three blocks are needed because each block can play the sound of a single note only. The musical notes on the staff are read from left to right and are transformed to blocks arranged from top to bottom as follows:

First note is E4 sound with 𝅝
Second note is G4 sound with ♩
Third note is C5 sound with 𝅘𝅥𝅯

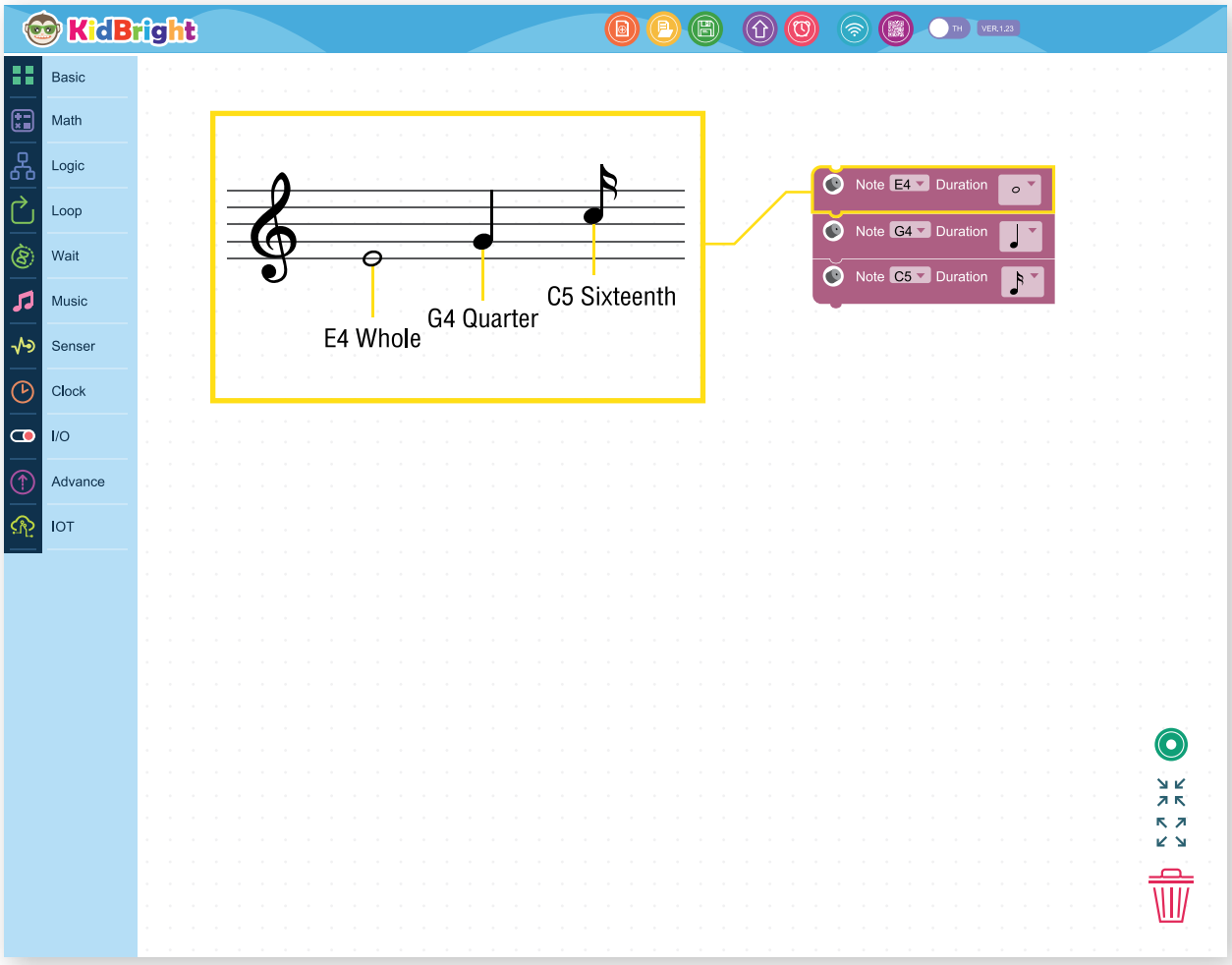


Figure 6.9 How to transform musical notes on a five-line staff to block codes.

Activity

Activity 6.1

When Switch 1 is pressed, the KidBright board plays sounds of notes C4, D4, E4 and shows the moving message of “Hello World!”.

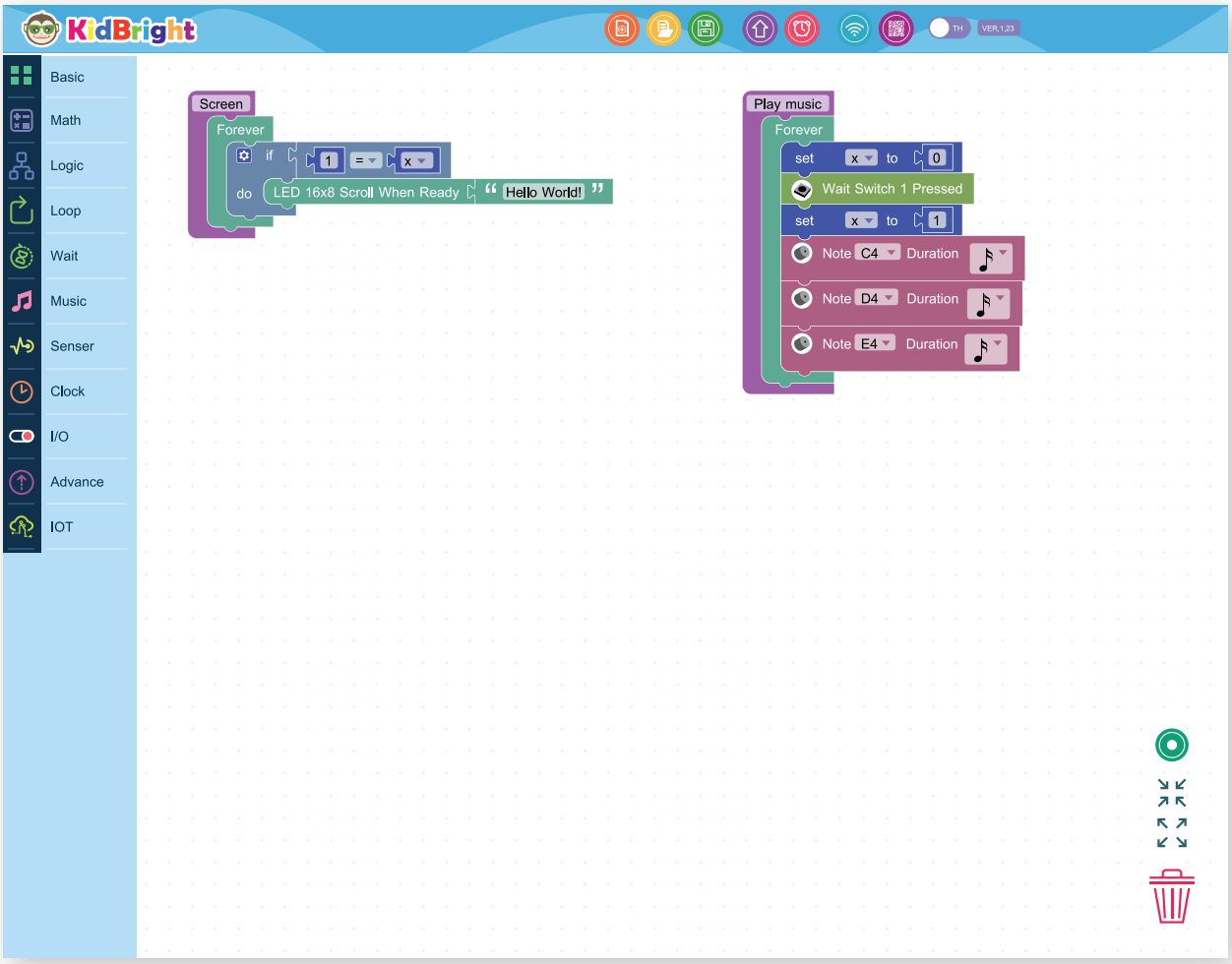


Figure 6.10 Program Activity 6.1.

How the program works

‘Screen’ task is a function for evaluating values of variable x, when variable x is equal to 1, the screen will display the message “Hello World”.

‘Music’ task responds with pressing the Switch 1. When “Switch 1” is pressed, the value of variable x will change from 0 to 1. The program will play sound of notes C4, D4, and E4, respectively. After that, the value of variable x will be set back to be 0 to be ready for the next action on the switch (pressed or not).

Repeat while

The block as shown in Figure 6.11 generates repeated functions. In this section, we will learn how to use the 'Repeat while' block, i.e., looping a specific part of a song.

Although some pieces of music can be long, there are sometimes sections where notes are repeated, e.g. a song chorus. We can cut down the number of coding commands needed by using the 'Repeat while' block.

The repeat while block has a repeating effect on the blocks that are in the boundary of the 'Repeat while' block, i.e., they will be repeated over and over when the given condition is true.

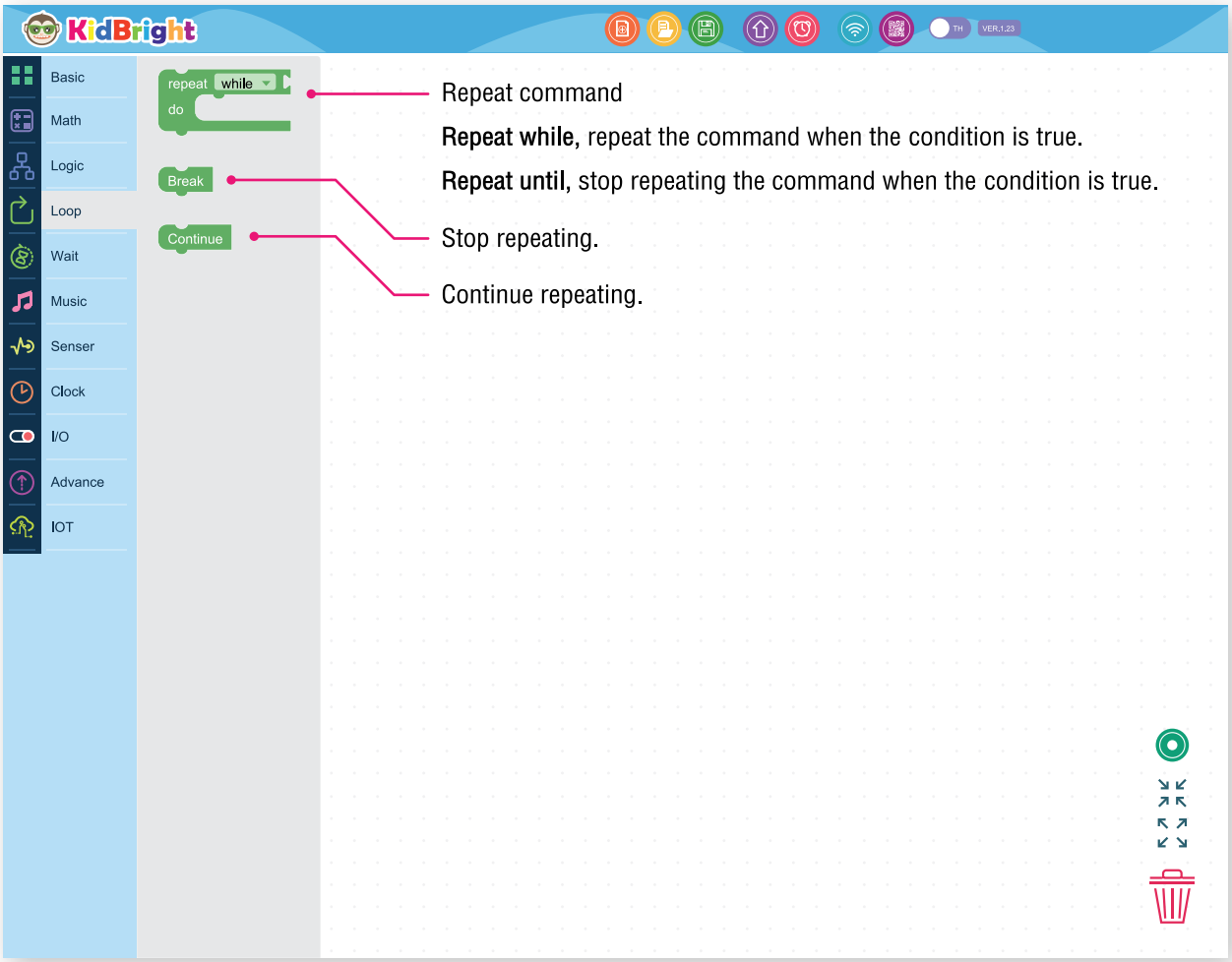


Figure 6.11 'Repeat while' block and assisting block.

Activity 6.2

The program consists of a set of coding commands to detect the status of Switch 1. If Switch 1 is pressed, the screen displays the word “Hi” until Switch 1 is released.

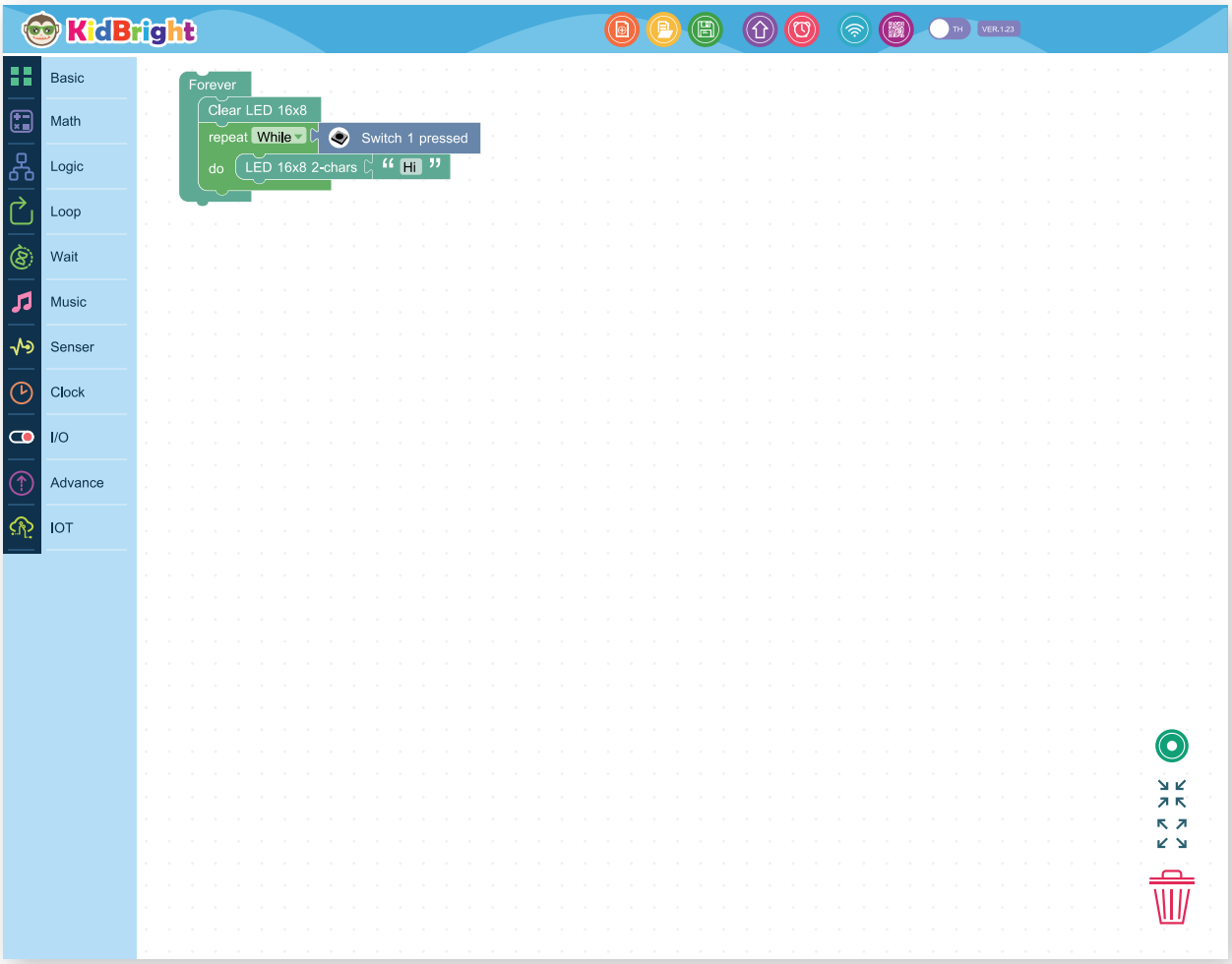


Figure 6.12 Program Activity 6.2.

How the program works

Letters shown on the LED screen are automatically deleted due to the command of Clear LED 16x8 block. However, when Switch 1 is pressed, the given condition of ‘Repeat while’ block will be true and the program will then work on the LED 16x8 2-chars “ Hi ” block, the word “Hi” is then shown on the LED screen until Switch 1 is released. Release of Switch 1 ends the process of the ‘Repeat while’ block since the given condition is false.

Activity 6.3

The program is a set of coding commands to detect the status of Switch 1. If Switch 1 is pressed, the program will play the sound of note C4 until Switch 1 is released.

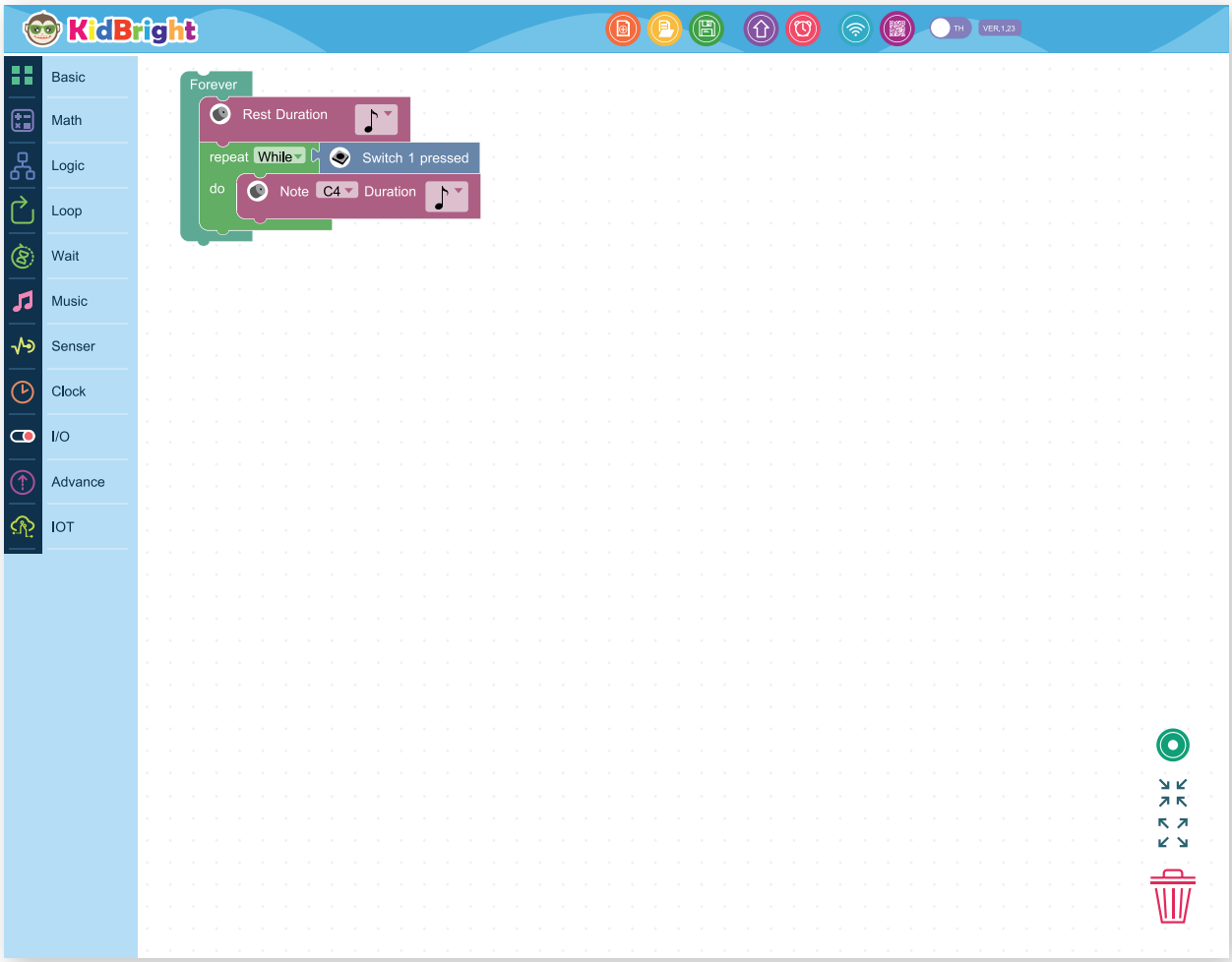
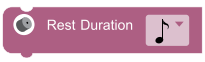
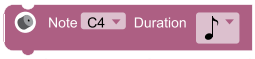


Figure 6.13 Program Activity 6.3.

How the program works

The buzzer is always silent due to the effect of  block. Whenever Switch 1 is pressed, the given condition of 'Repeat while' block is true and the program executes  block. Block which unmutes the buzzer and the sound of note C4 is played. When the Switch 1 is released, the process of the 'Repeat while' block ends since the given condition is false.

Activity 6.4

The program is a set of coding commands to detect the status of Switch 1. If the Switch 1 is pressed, the program will play the sound of note C4 twice.

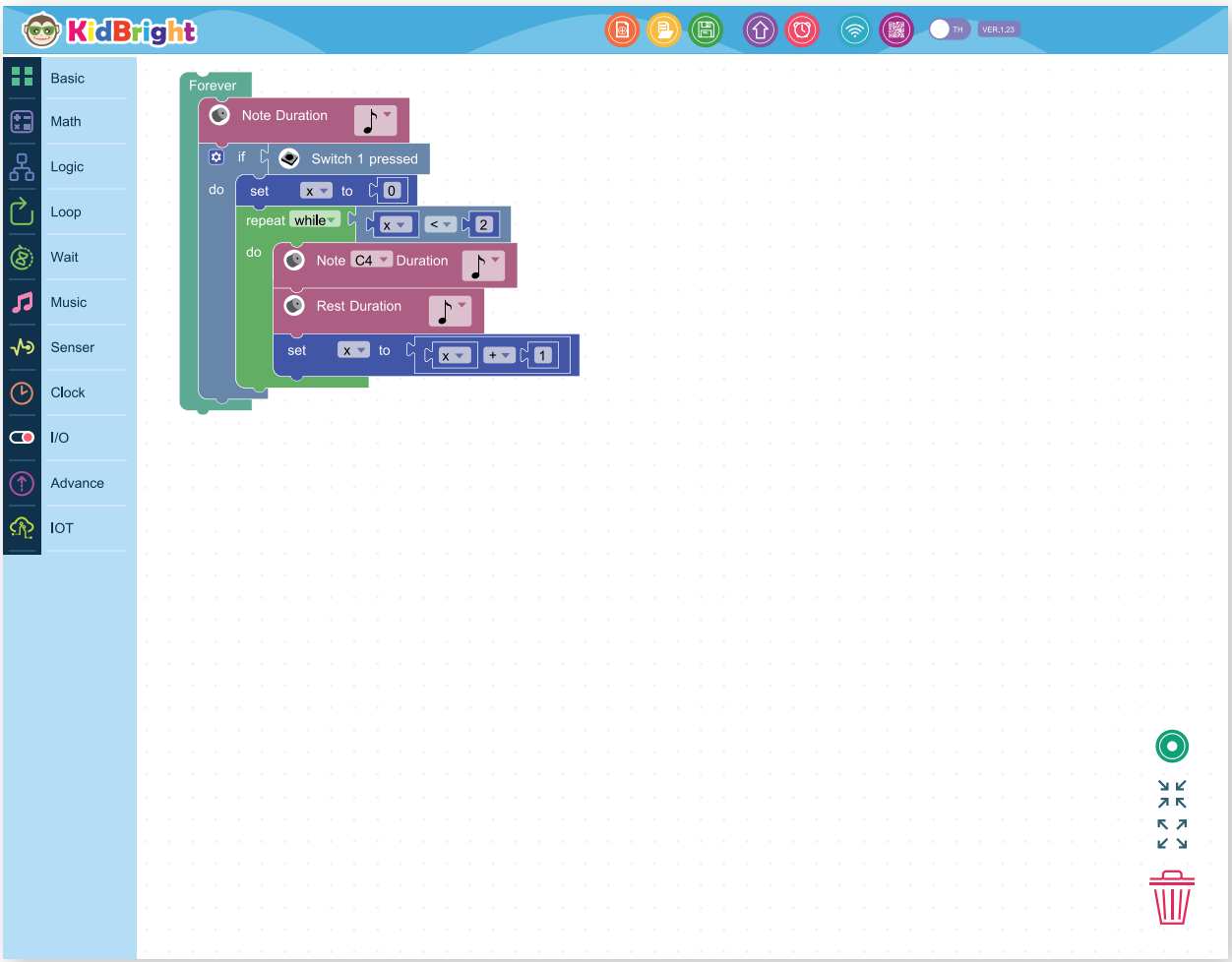


Figure 6.14 Program Activity 6.4.

How the program works

The buzzer is silent with the effect of 

block. Whenever Switch 1 is pressed, the program will use variable x to count how many times that a note is played. With the command of the 'Repeat while' block, the note will be played repeatedly. However, each time the note is repeated, the value of variable x increases by one. If the note is repeated twice, the value of variable x is two. When the program checks the condition of x<2, the condition becomes false. This ends the repeat function and the buzzer becomes silent.

Activity 6.5

The program is a set of coding commands to detect the status of Switch 1. When Switch 1 is pressed, the program will play the sound of note C4 repeatedly until Switch 2 is pressed.

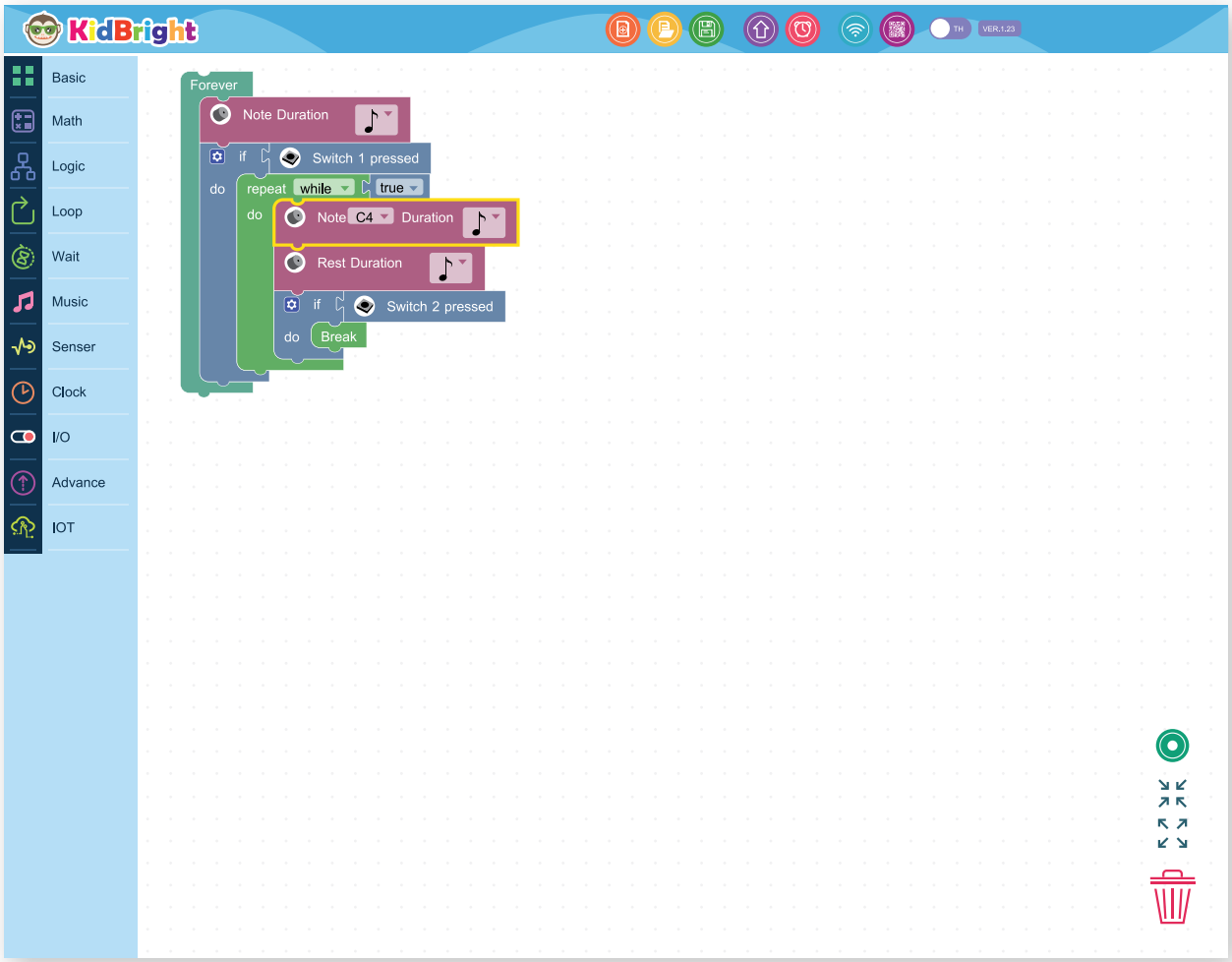
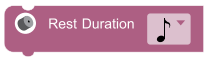


Figure 6.15 Program Activity 6.5.

How the program works

The buzzer is silent because the 

block is activated. Whenever Switch 1 is pressed, the program executes coding commands on the 'Repeat while' block which makes the buzzer play the sound of note C4 at intervals separated by eighth period rests. This process will repeat as the given condition of the 'Repeat while' block is true. If Switch 2 is pressed, the function of 'Repeat while' block ends and the commands of the 

block will be in action.

‘Wait’ block

This function is used to wait for Switch 1 to be pressed or released, Switch 2 to be pressed or released, or the display screen to be ready. The wait blocks are arranged in the Wait tab as shown in Figure 6.16.

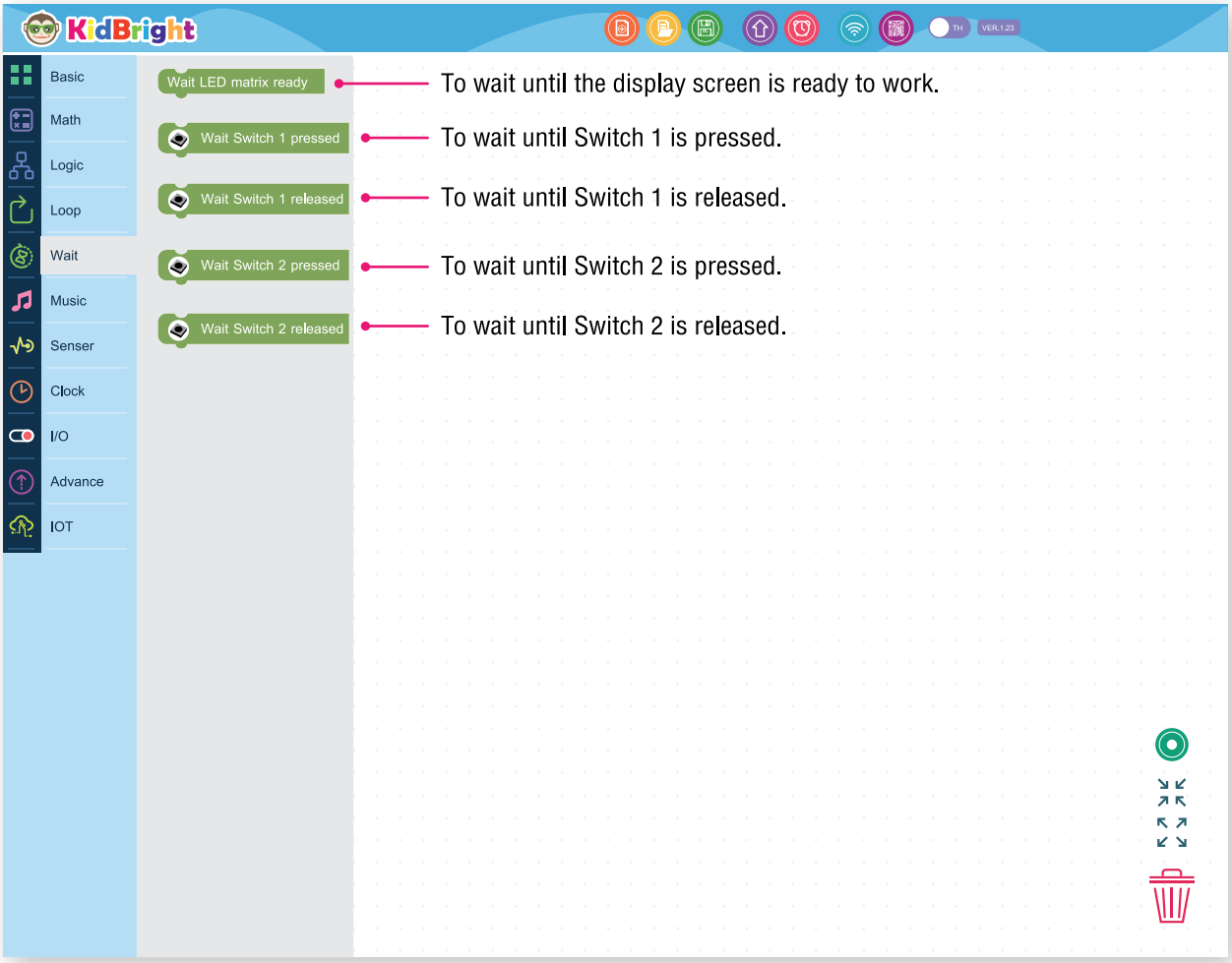


Figure 6.16 Blocks in Wait tap.

Activity 6.6

The program consists of a set of coding commands to wait for Switch 1 to be pressed and then released. Then the sound of note C4 is played before the process goes back to the waiting stage.

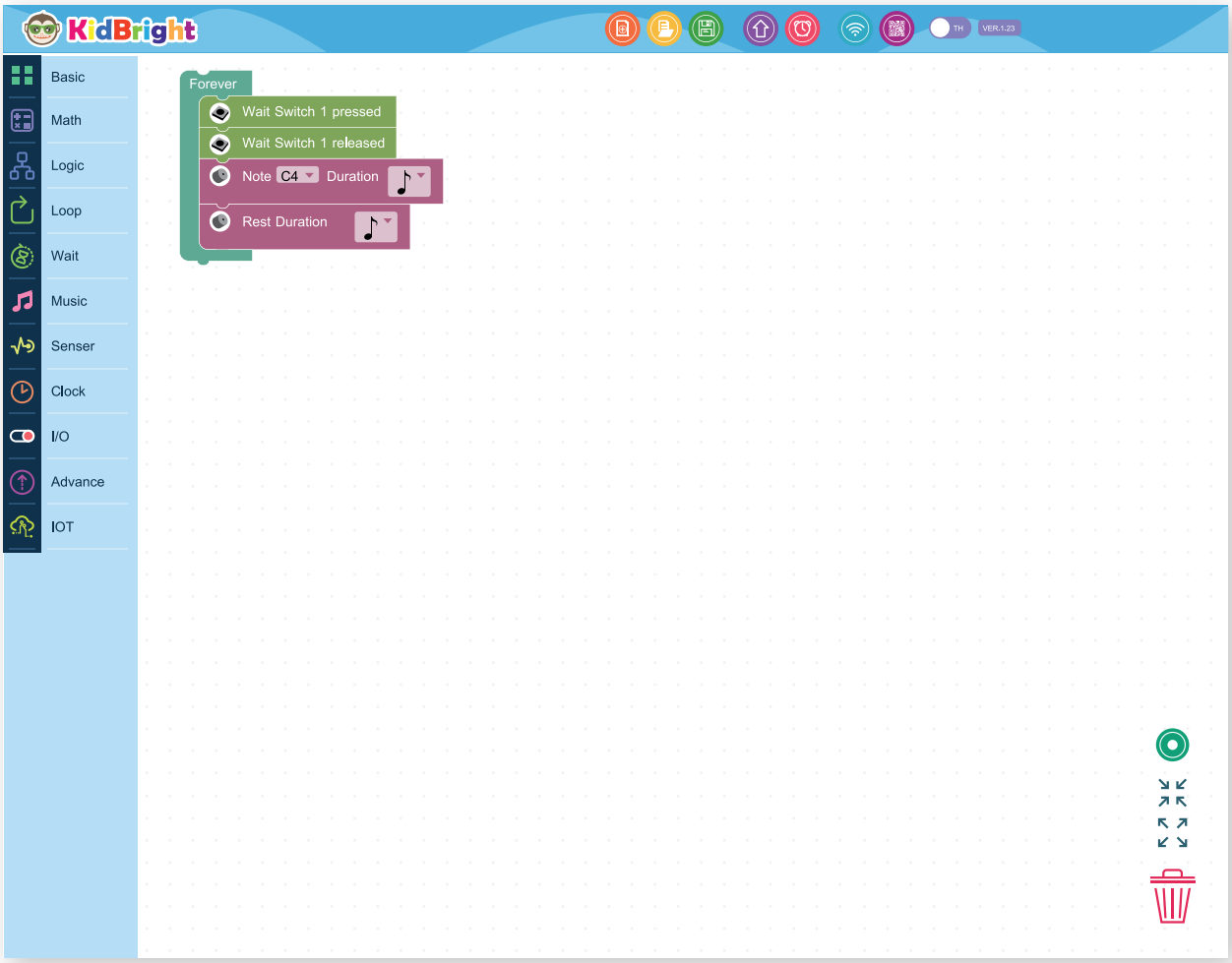


Figure 6.17 Example of Activity 6.6.

Explaining how the program works

First, the program begins with waiting for the Switch 1 to be pressed. When the Switch 1 is pressed, the program will wait until it is released. After the Switch 1 is released, the program then plays note C4 once, before going back to the waiting stage.

Activity 6.7

Program to play a classical song.

Loy Krathong Song

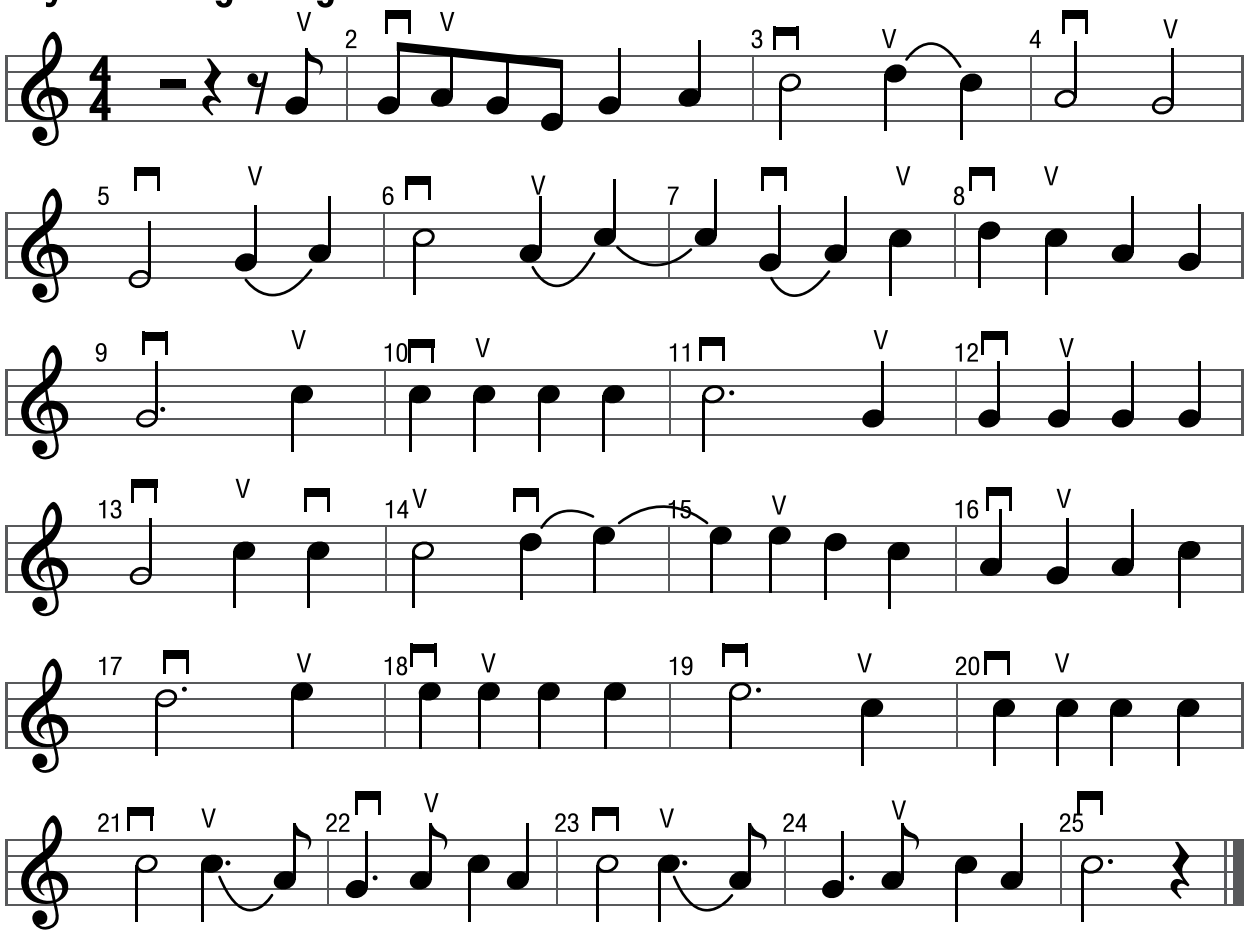


Figure 6.18 The classical Thai song “Loy Krathong” in staff notation.

The program consists of a set of coding commands that waits for Switch 1 to be pressed before the Loy Krathong song is played.

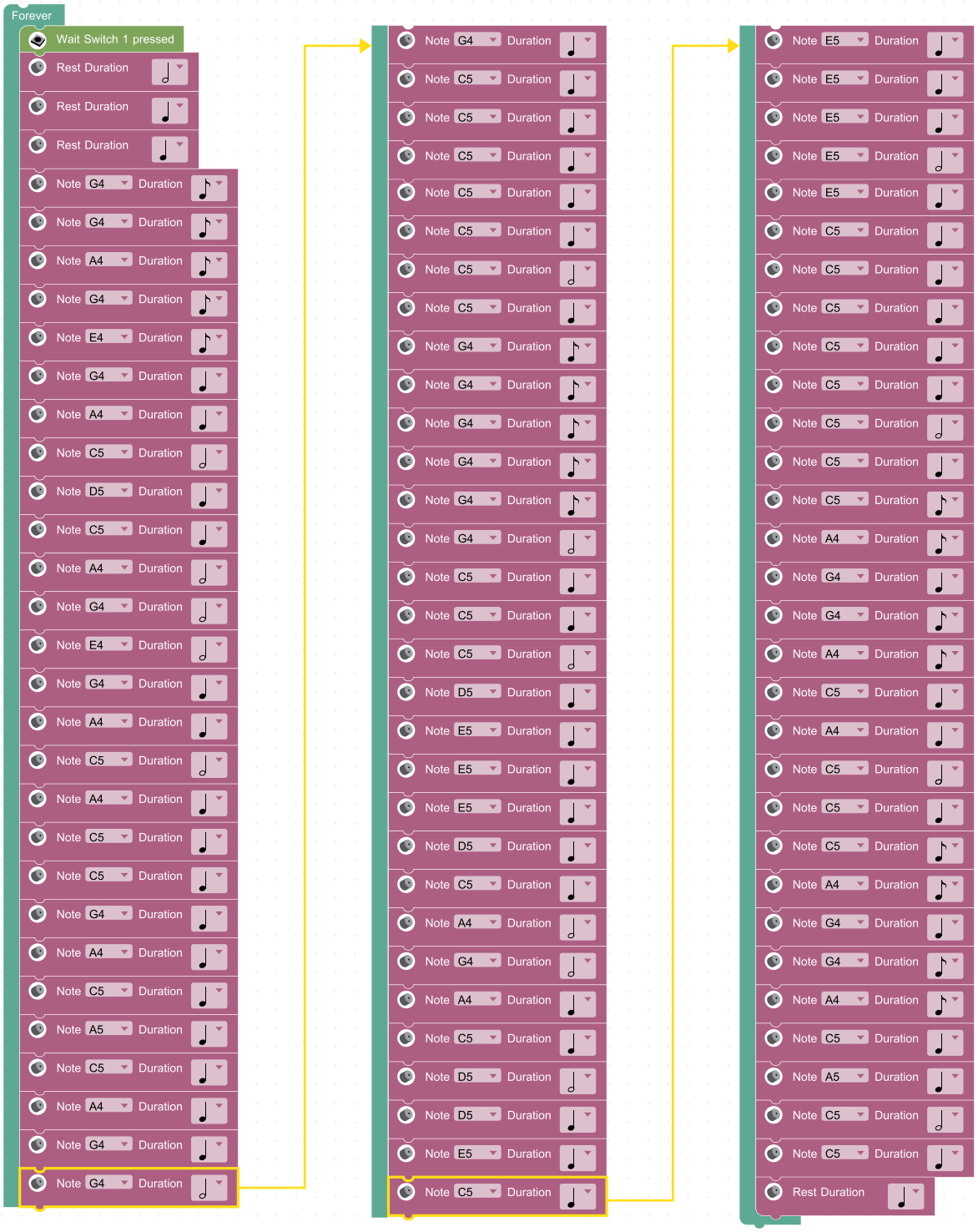


Figure 6.19 "Loy Krathong" song program.

Explaining how the program works

When the Switch 1 is pressed, the program will play Loy Krathong song.

Activity 6.8

Program to play Phon Pi Mai Song (New Year Greetings)

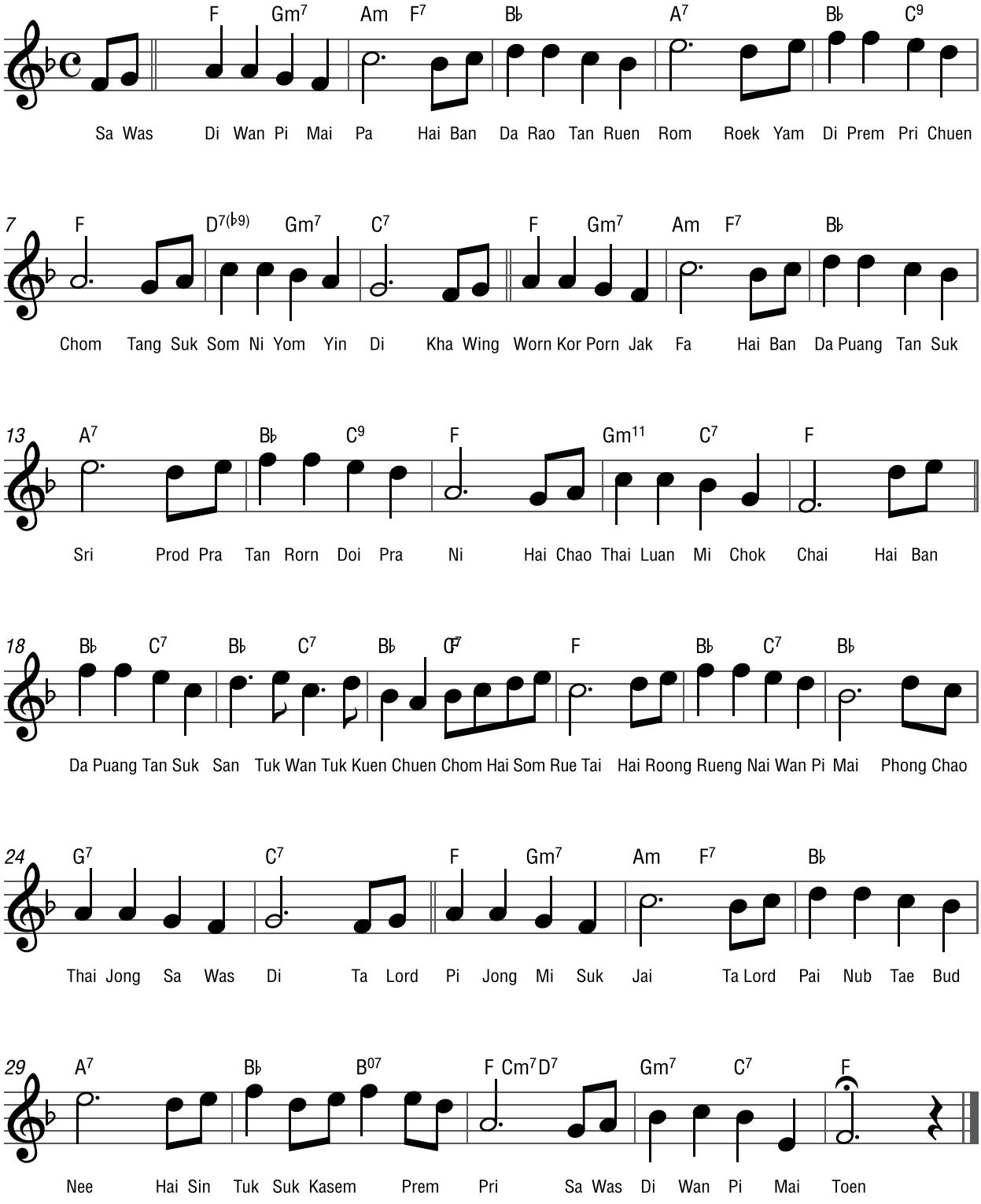
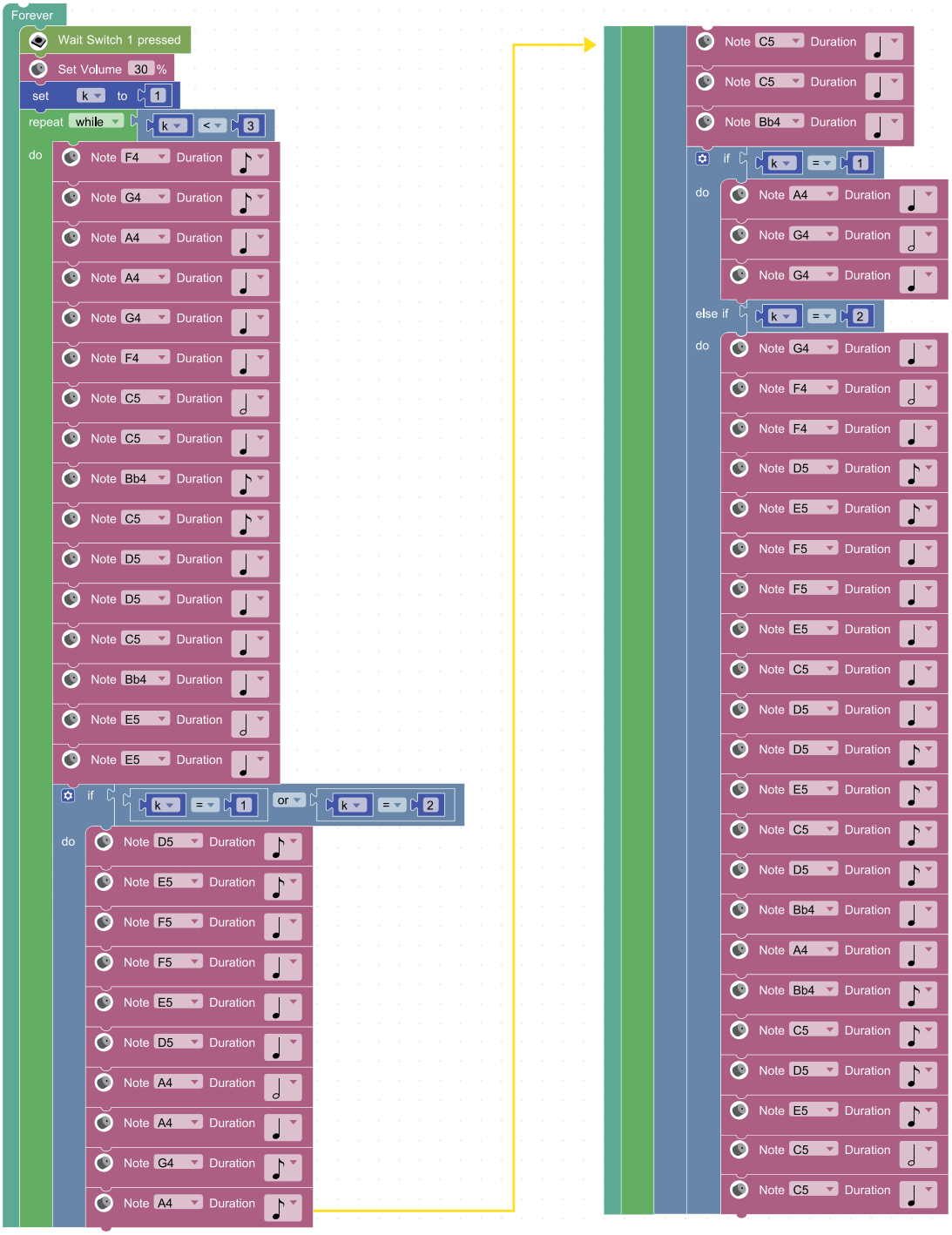


Figure 6.19 Phon Pi Mai song note.

With the following program, when Switch 1 is pressed, the song Phon Pi Mai song will be played.



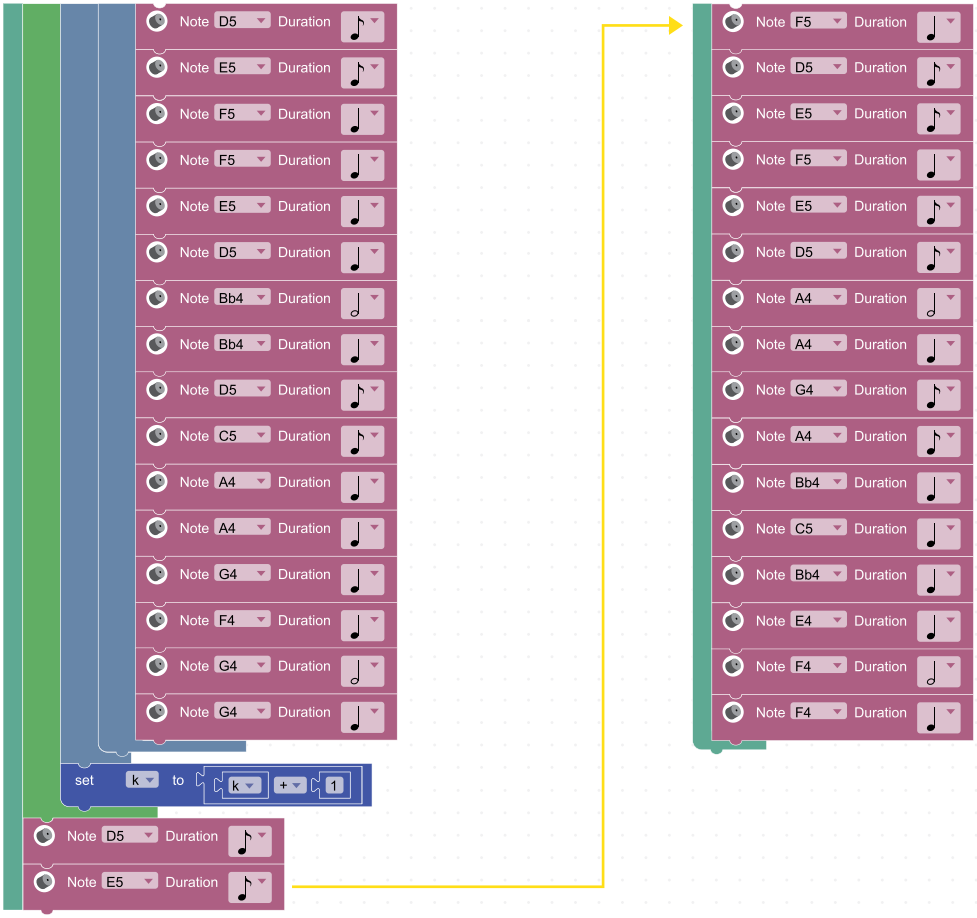


Figure 6.21 Phon Pi Mai song program.

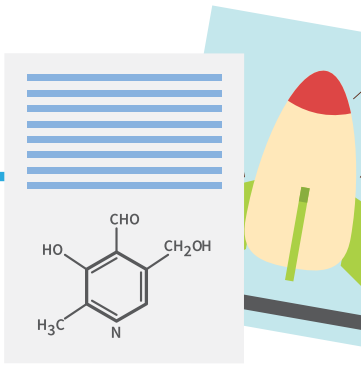
How the program works

When the Switch 1 is pressed, the program will play Phon Pi Mai song. The 'Repeat while' block is used for some of repetitive sections of the song to shorten the number of coding commands

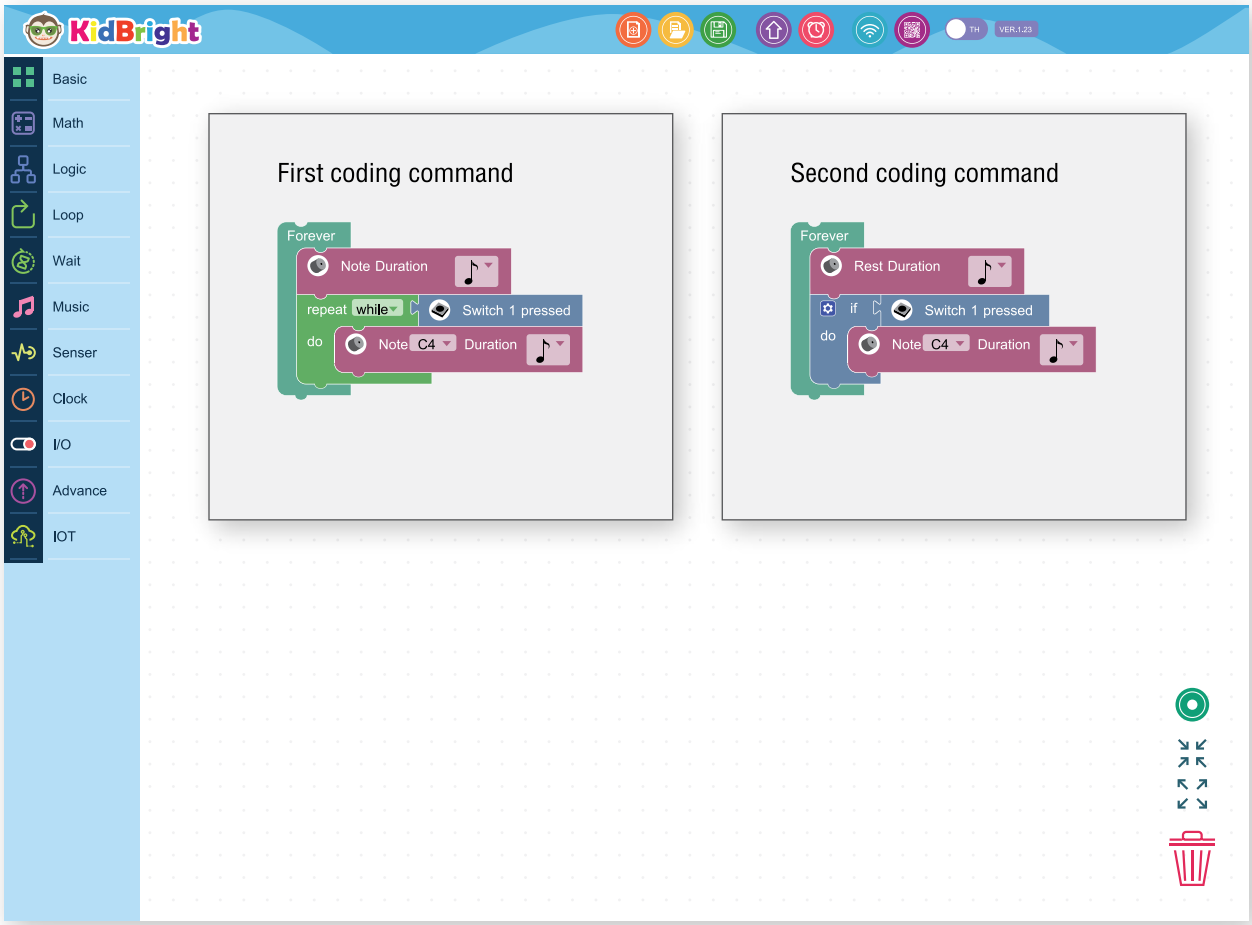
Summary

Students learn music blocks and and how to use them in the Music Tab for composing music.

Exercise

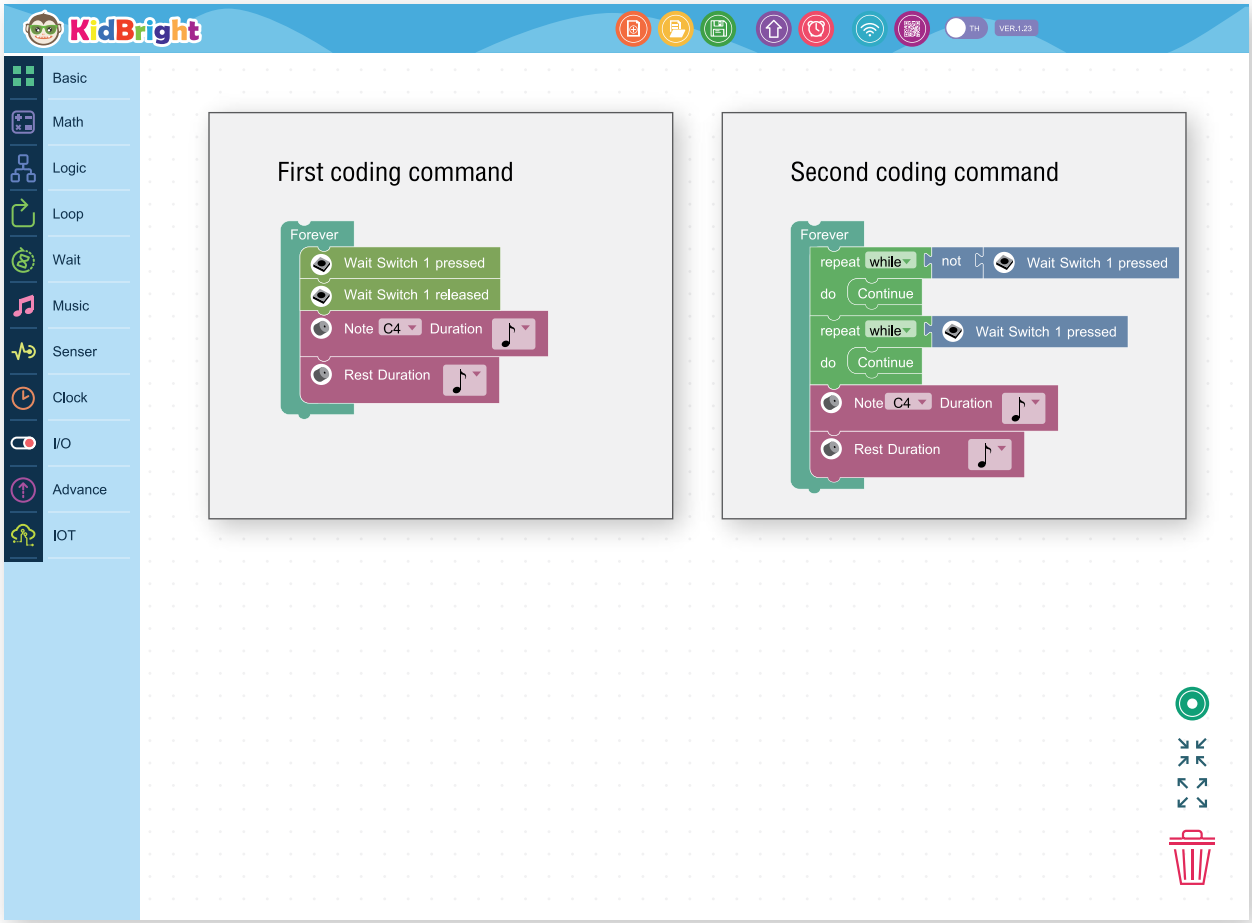


1. Compare the following sets of coding commands. How are they different?



2. Predict the results of the first and second set of coding commands.

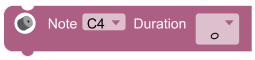
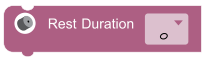
3. Test your predictions by running the two sets of coding commands and note how they are different.

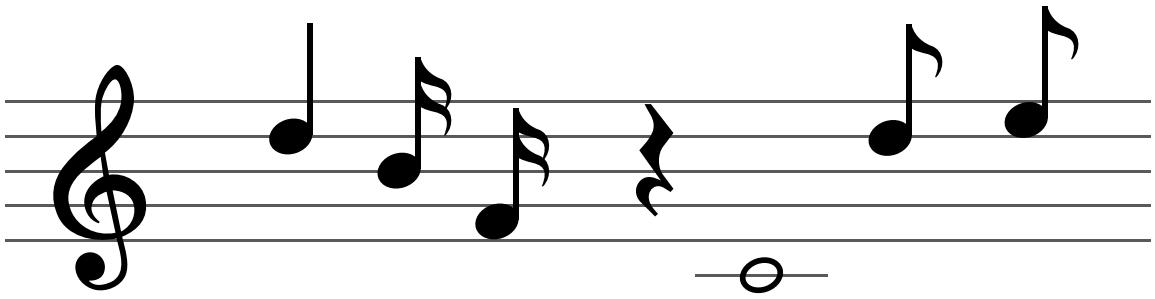


4. Fill in the names of the musical notes and rhythm (number of beats) of each note below.

Note symbols	English name	Rhythms (4/4time signature)
		
		
		
		
		

5. Describe the functions of the 'Sounding' block and the 'Resting' block.

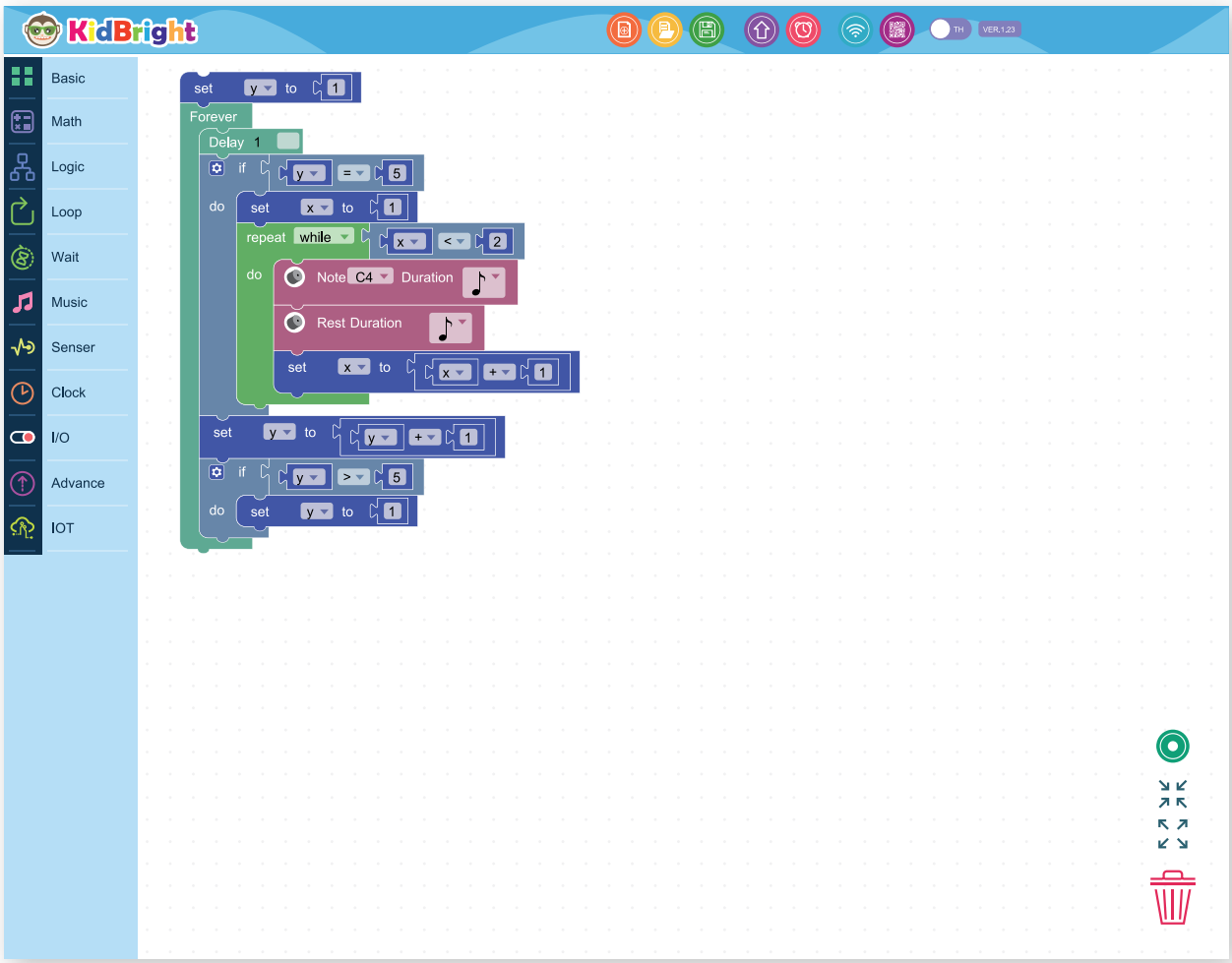
	
	



6. Use ‘Sounding’ blocks and ‘Resting’ blocks to develop coding commands to play the musical notes on the five-line staff below.

7. Add a ‘Repeat while’ block in the coding commands to play the notes repeatedly.

8. In the following coding commands, for how many seconds does the buzzer play a loop of a note C7 twice?



9. Create a command series to play the “Happy Birghtday” song.

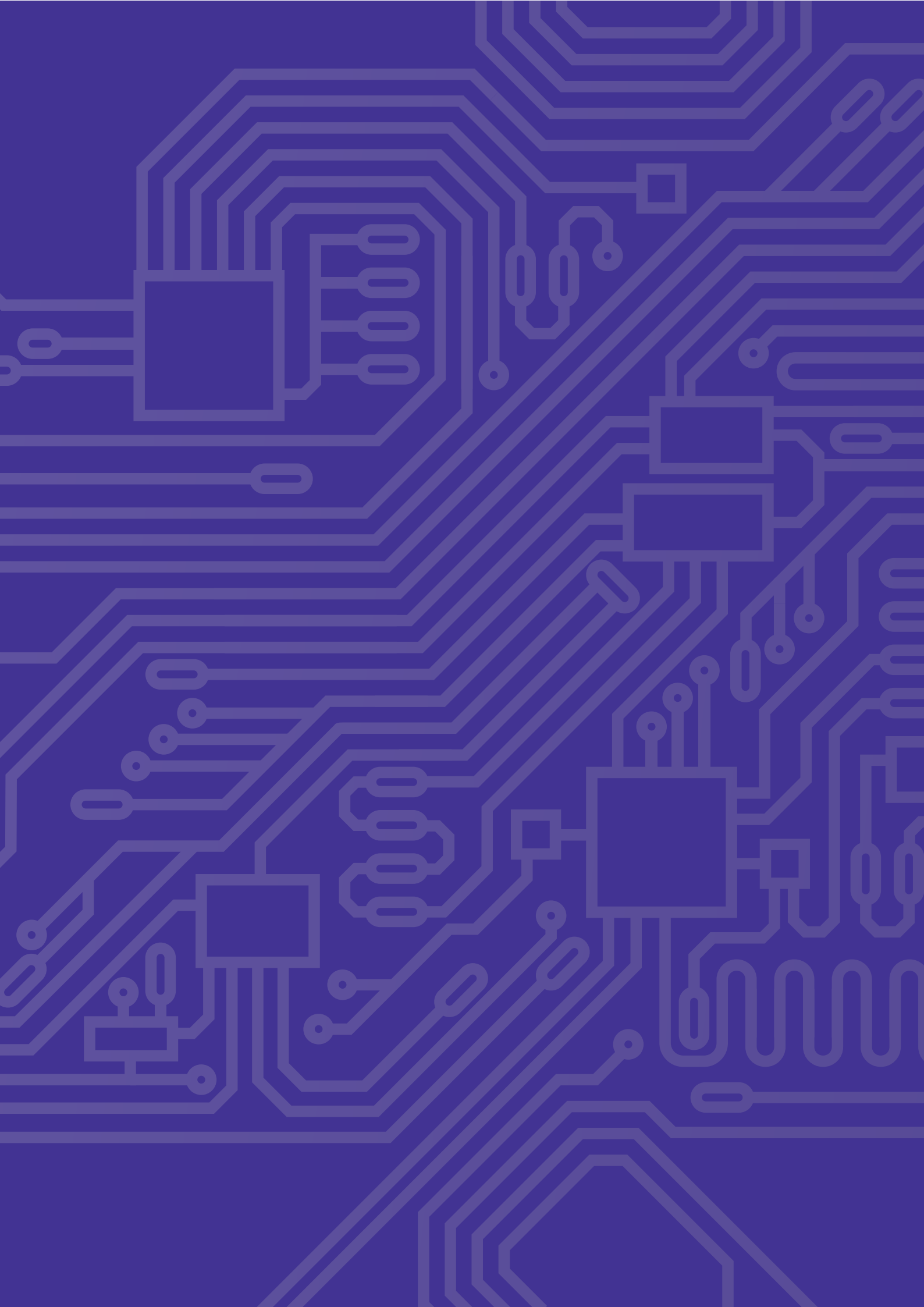
Chapter 7

Play with the Clock

Learning Objectives

At the end of the learning process of Chapter 7, students will be able to:

1. Understand blocks on the Clock tab.
2. Understand the concept of mulitasking.
3. Use the 'Task' block.
4. Implement a timer program by using blocks on the Clock tab.



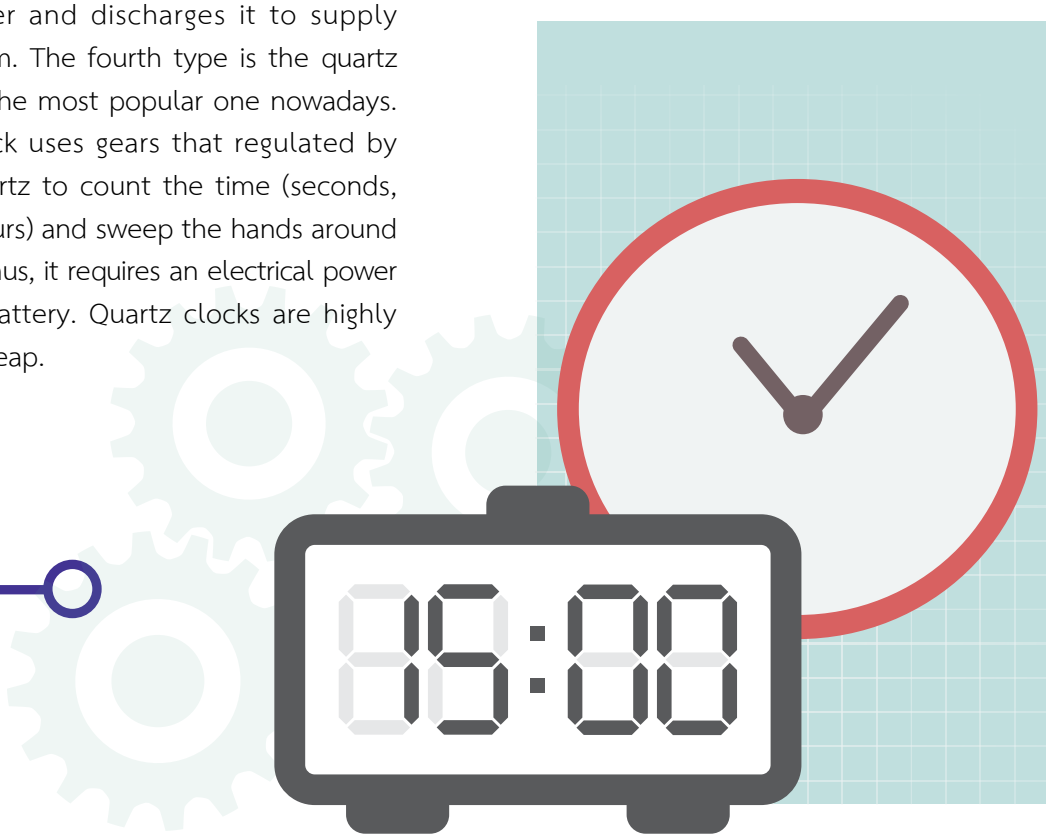
Learning Content

How clocks work

There are four types of clock in common use. A winding-type clock has a spiral torsion spring of metal ribbon, called the mainspring, which is used as a power source in the clock. In order to keep time, the user must wind the mainspring regularly. The winding-type clock is not used much today because it is difficult to use properly. The second type is the automatic watch, which was developed from the winding clock. The automatic clock is typically a watch worn on the wrist and is powered by the natural movement of the wearer. A weight or a pendulum inside the automatic watch moves with the movement of the wearer that winds the mainspring of the clock. The third type is the kinetic clock, which is a combination of the automatic clock and the winding clock. Instead of winding the mainspring by its weight, the kinetic clock stores electrical power and discharges it to supply the clock system. The fourth type is the quartz clock, which is the most popular one nowadays. The quartz clock uses gears that regulated by a crystal of quartz to count the time (seconds, minutes and hours) and sweep the hands around the clockface. Thus, it requires an electrical power supply, e.g. a battery. Quartz clocks are highly accurate and cheap.

Elements used in coding a date-and-time displaying program

Figure 7.1 shows all blocks on the Clock tab, namely Date/Time, Date, Time, Day, Month, Year, Hour, Minute, and Second. The KidBright displays dates and times on the 'LED 16x8 Scroll When Ready' block because the area of the LED display device is not wide enough to show the date and time simultaneously. Therefore, the text is scrolled from right to left letter-by-letter.



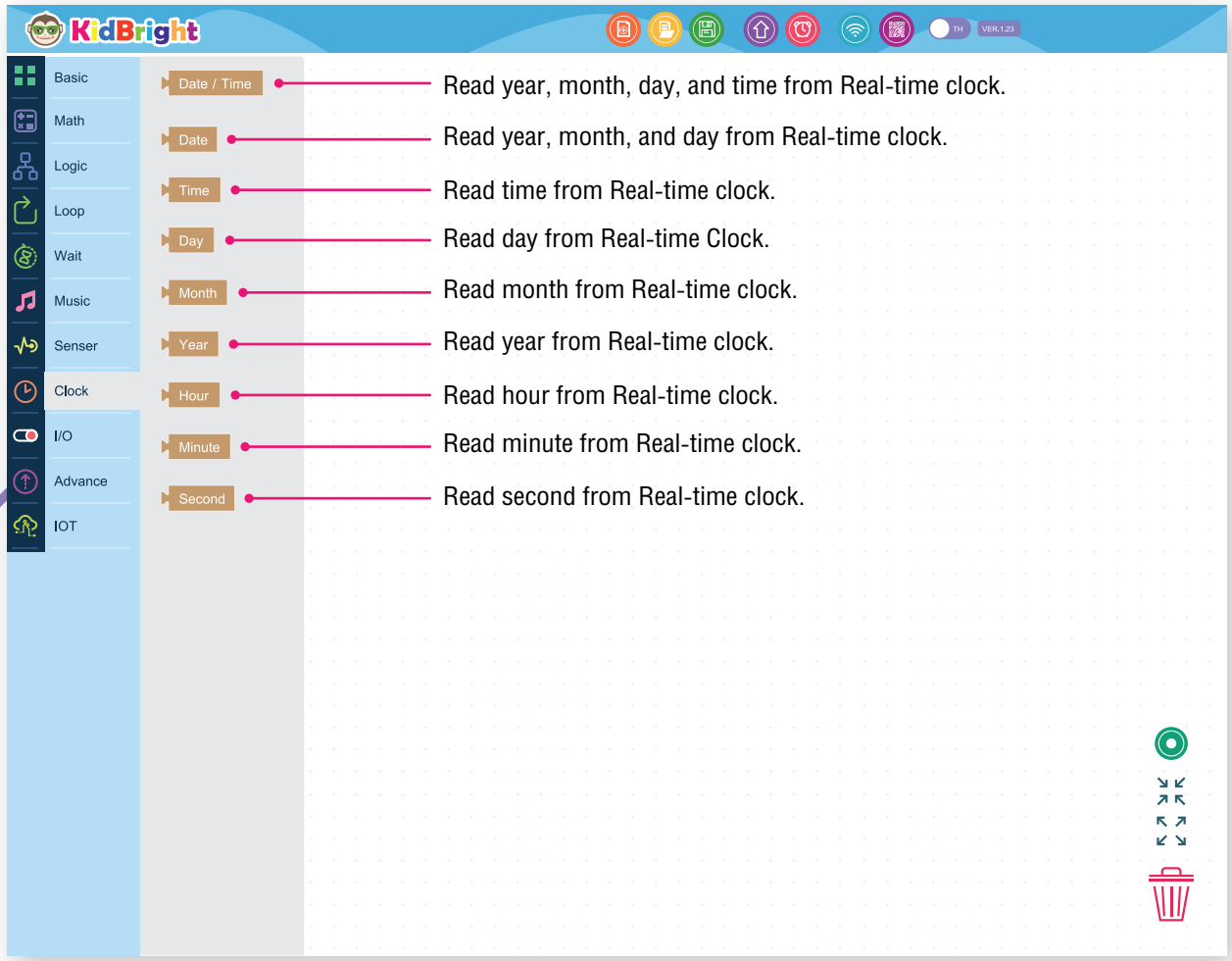


Figure 7.1 Blocks on the Clock tab.

Based on data types returned by the blocks on the Clock tab, these blocks can be categorized into two groups.

The first group includes all blocks that return characters, i.e., Date/Time, Date, and Time. These blocks are suitable for displaying on the 'LED 16x8 Scroll When Ready' block.

The second group includes those that return numbers, i.e., Day, Month, Year, Hour, Minute and Second. These blocks can be used in mathematical calculations such as time comparison and date comparison.

How to use blocks on the Clock tab?

The KidBright board is equipped with a device called a Real-time clock. This device is useful in many science/engineering projects that require an automatic control of process with a time schedule.

The basic operation of the Real-time clock

When the KidBright board is connected to a DC power source, the Real-time clock is powered and the date and time are set to the initial values of 30/04/2018 and 00:00, respectively. To set the date and time, click the **Set Clock** button. The Set Clock box with a button that shows the current date and time as shown in Figure 7.2, will appear. Click that button and then click **OK** to set that date and time to the Real-time clock, and the Real-time clock will operate continuously as long as the DC power supply is connected to the KidBright board.

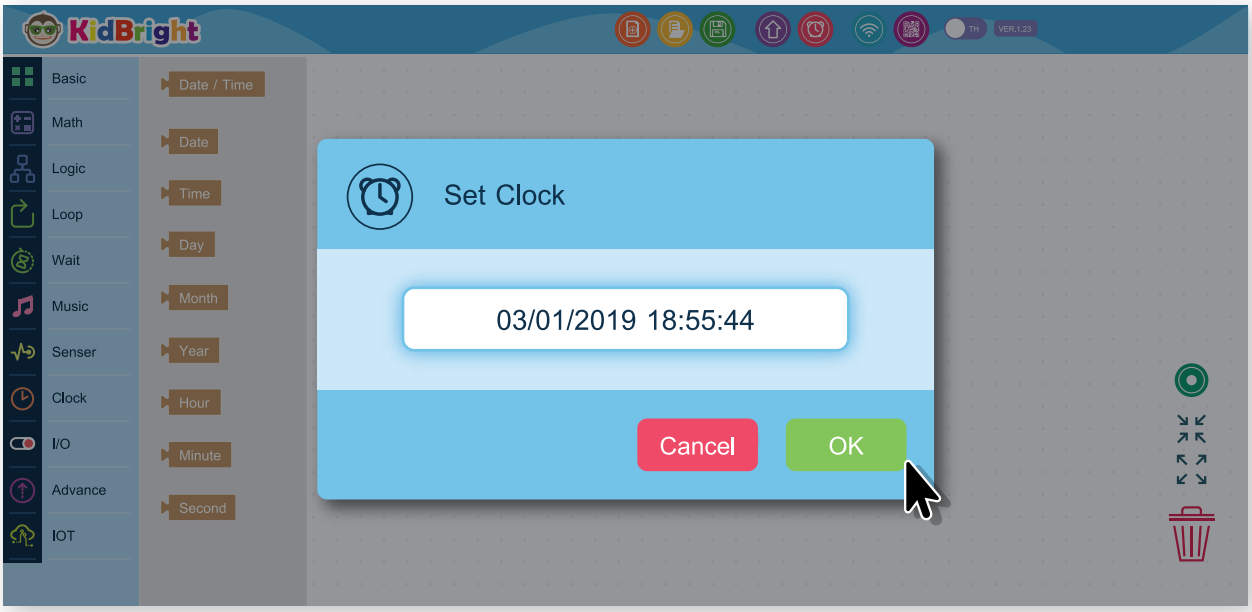


Figure 7.2 Clock setting.

When the KidBright board is disconnected from the power source, the date and time of the Real-time clock are reset to the initial values. As a consequence, the Real-time clock needs to be reset. In order to keep the date and time without reset after each use, a battery must be put into the battery socket on the back side of the KidBright board, as shown in Figure 7.3.

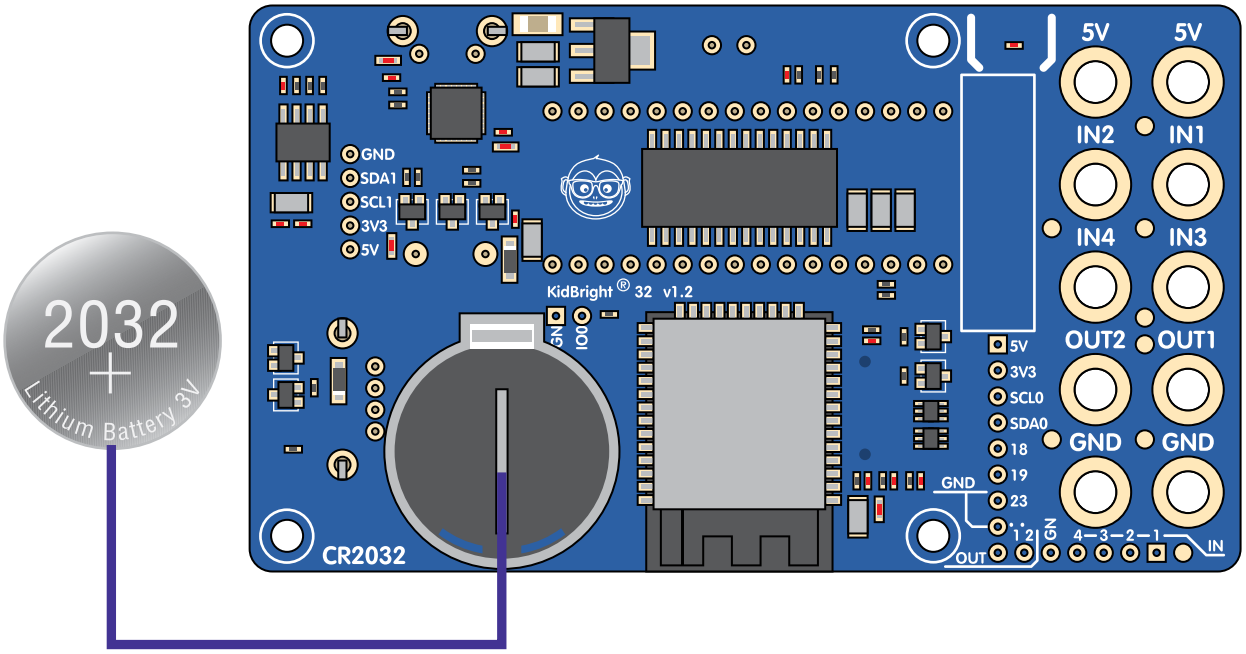
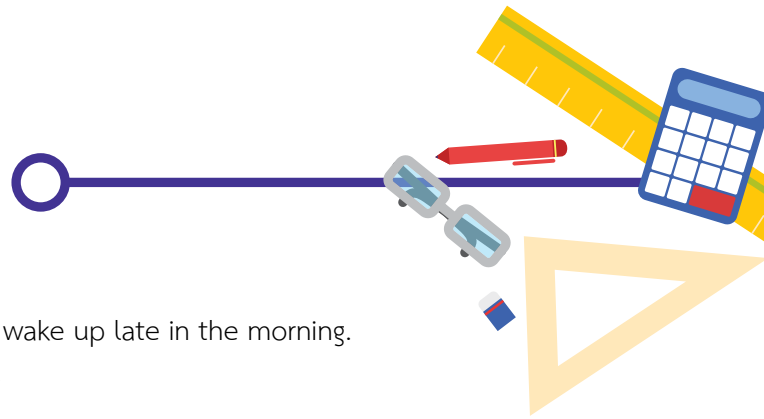


Figure 7.3 Battery socket on the back side of the KidBright board.



Activity

Activity 7.1

You have an exam on the next Monday, and you often wake up late in the morning. How can you solve this problem? Explain your method.

Activity 7.2

Displaying the date and time on the ‘LED 16x8 Scroll When Ready’ block.

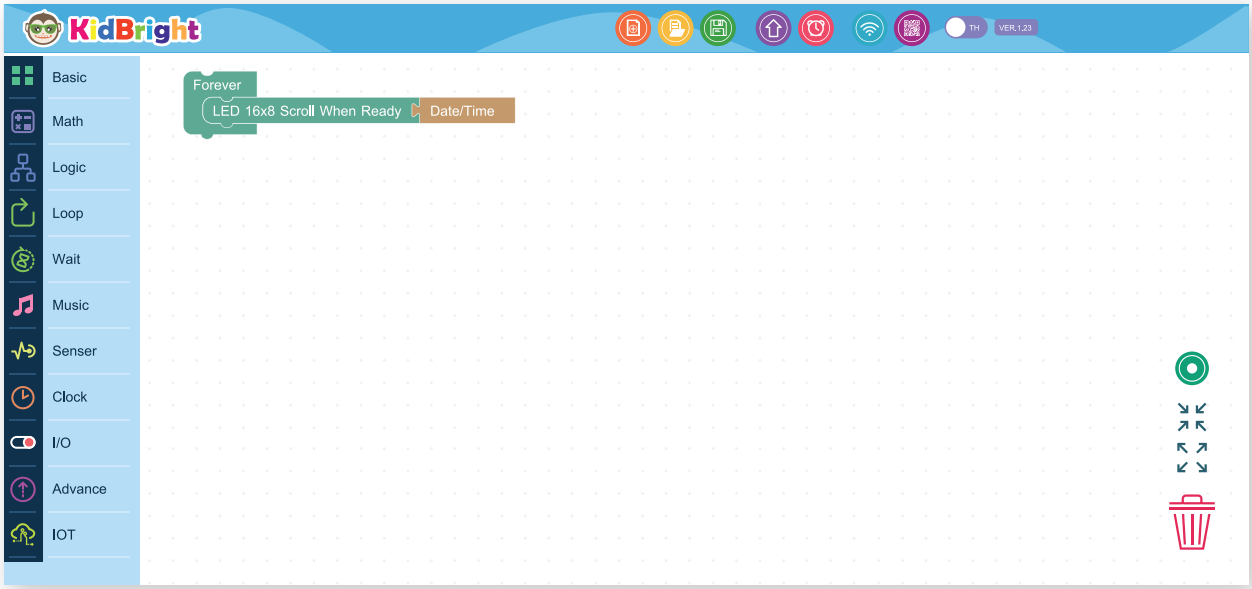


Figure 7.4 Program for displaying the date and time.

Description of program in Figure 7.4.

Displaying the time using the ‘LED 16x8 Scroll When Ready’ and ‘Date/Time’ blocks has the following details.

1. Set a forever loop.
2. Display the date and time on the ‘LED 16x8 Scroll When Ready’ block.
3. Go to step 2.

Remark Set the date and time by clicking the **Set Clock** button to change the date and time of the Real-time clock to the current date and time.

Result

When the command blocks are compiled into the machine code and then transferred to the board, the date and time will run on the screen from right to left continuously as shown in Figure 7.5

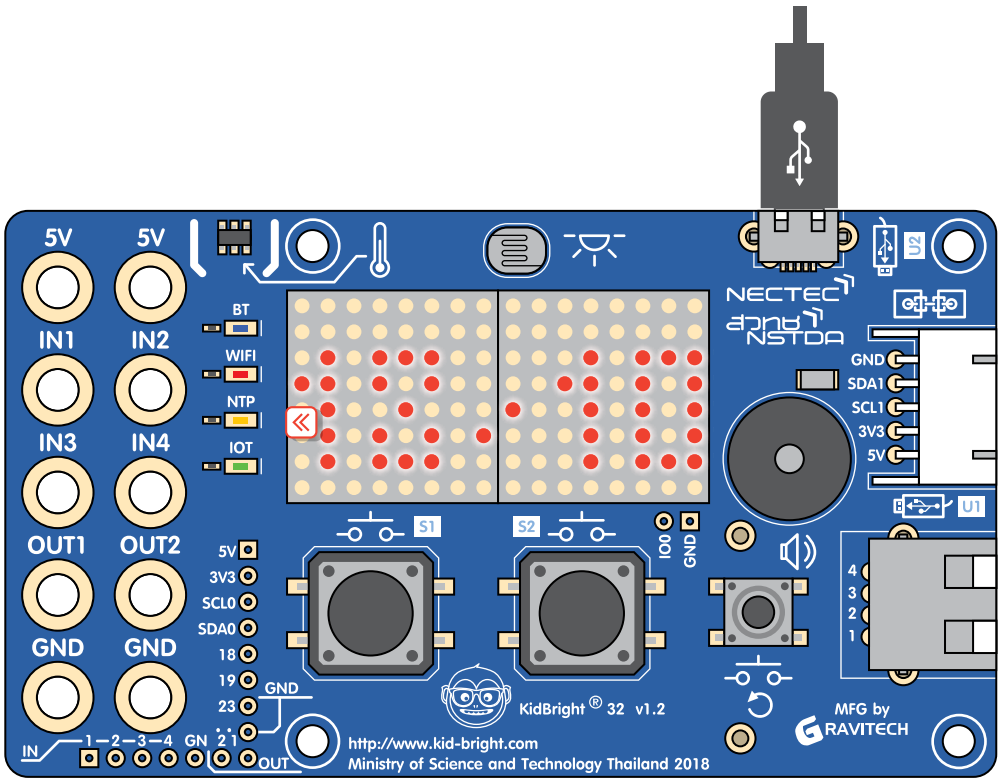


Figure 7.5 Displaying the date and time on the screen of the KidBright board.

Activity 7.3

Displaying the time using the 'LED 16x8 Scroll When Ready' block.

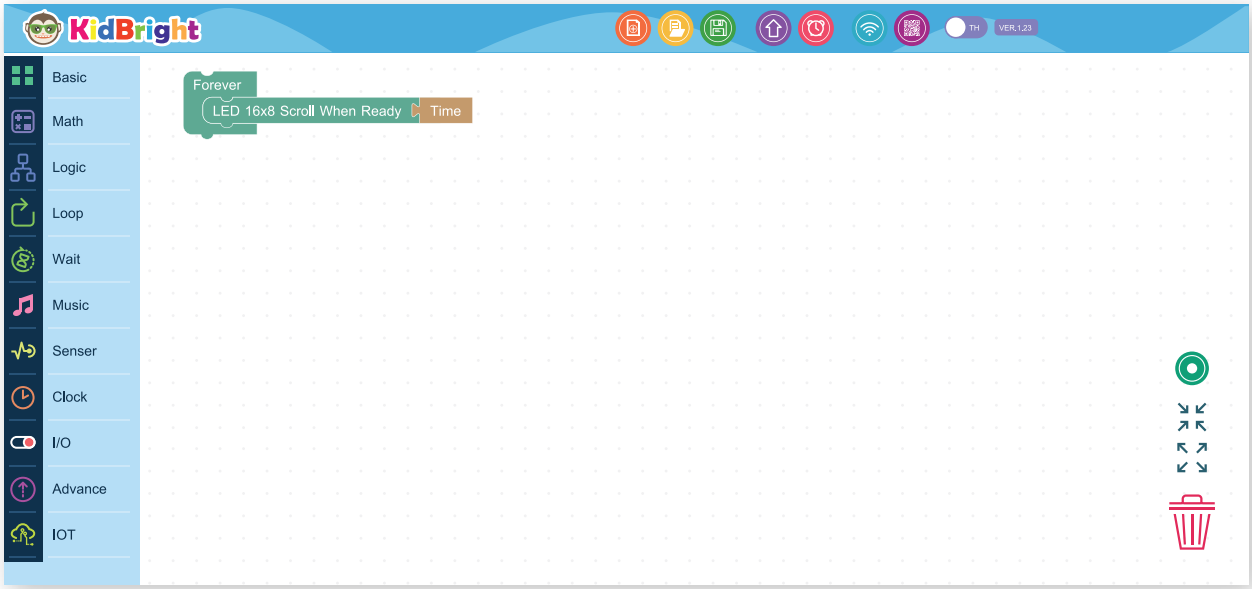


Figure 7.6 Program for displaying the time.

Description of program in Figure 7.6.

Displaying the time using the 'LED 16x8 Scroll When Ready' and 'Time' blocks has the following details.

1. Set a forever loop.
2. Display the time on the 'LED 16x8 Scroll When Ready' block.
3. Go to step 2.

Result

The time will run on the screen from right to left continuously.

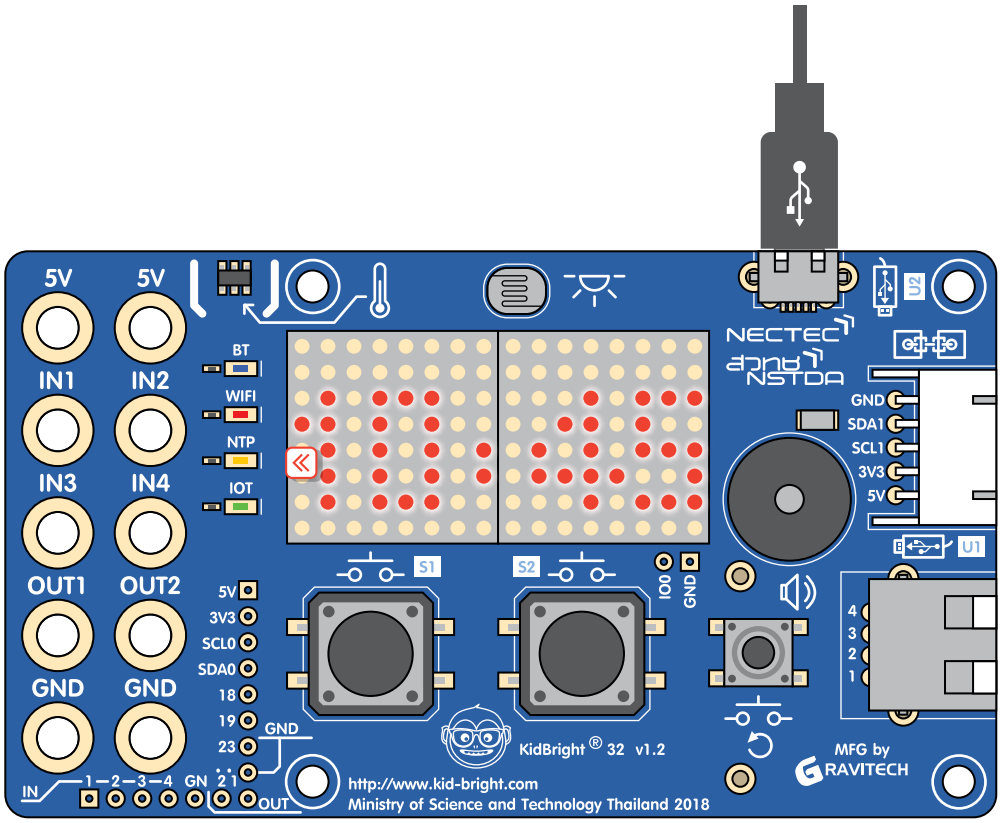


Figure 7.7 Displaying the time on the screen of the KidBright board.

Activity 7.4

Displaying the second on the screen of the Kidbright board using 'LED 16 ×8 2-char' and 'Second' blocks.

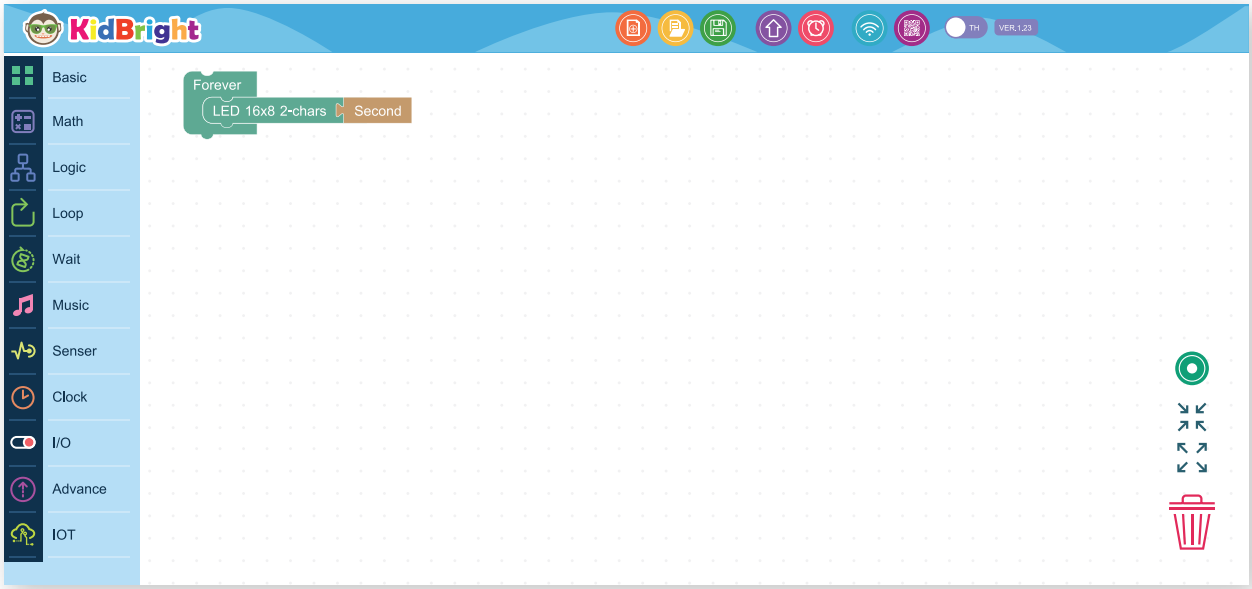


Figure 7.8 Program for displaying the second on the 'LED 16x8 Scroll When Ready'.

Description of program in Figure 7.8.

Displaying the second using the 'LED 16x8 2-chars' and 'Second' blocks has the following details.

1. Set a forever loop.
2. Display seconds on the 'LED 16x8 2-chars' block.
3. Go to step 2.

Result

In Figure 7.9, the seconds value will be displayed on the LED screen without moving. This value increases every second, starting from 0 to 59. When the value reaches 59, it will return to 0.

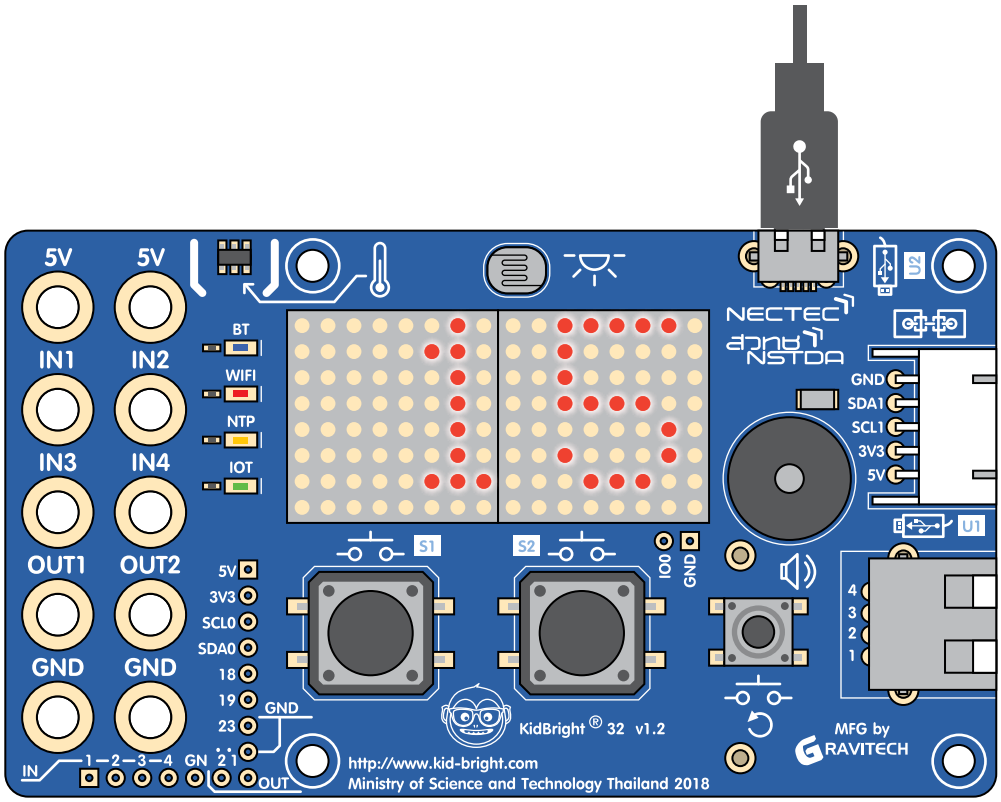


Figure 7.9 Displaying the second on the screen of the KidBright board.

Activity 7.5

Writing a program for playing a sound when the seconds is equal to 10.

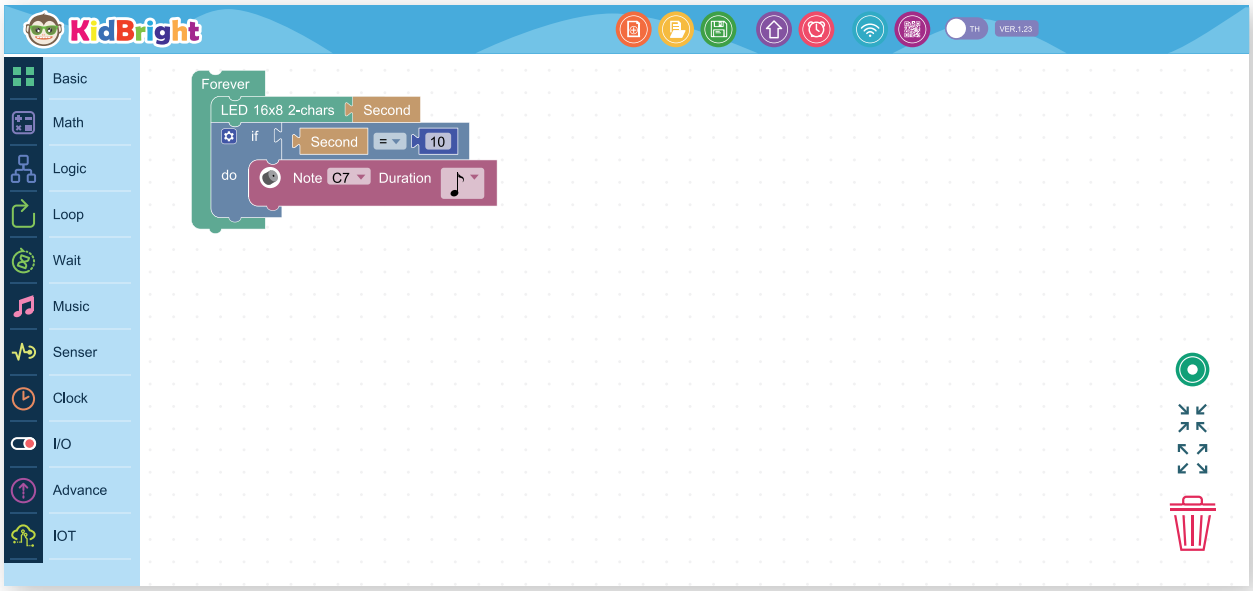


Figure 7.10 Program that plays a sound when the seconds value is equal to 10.

Description of program in Figure 7.10.

The program for playing a sound when the seconds value is equal to 10 has the following details.

1. Set a forever loop.
2. Display the value of the second on the 'LED 16x8 2-chars' block.
3. Check the following condition : Is it ture that "the value equals 10"?
   3.1`If the condition is ture, the note C7 is played once.
4. Go to step 2.

Result

The seconds value will be displayed on the LED screen. This value increases every second, starting from 0 to 59. When the value equals 10, the note C7 will be played.

Activity 7.6

Write an alarm clock program as shown in Figure 7.11.

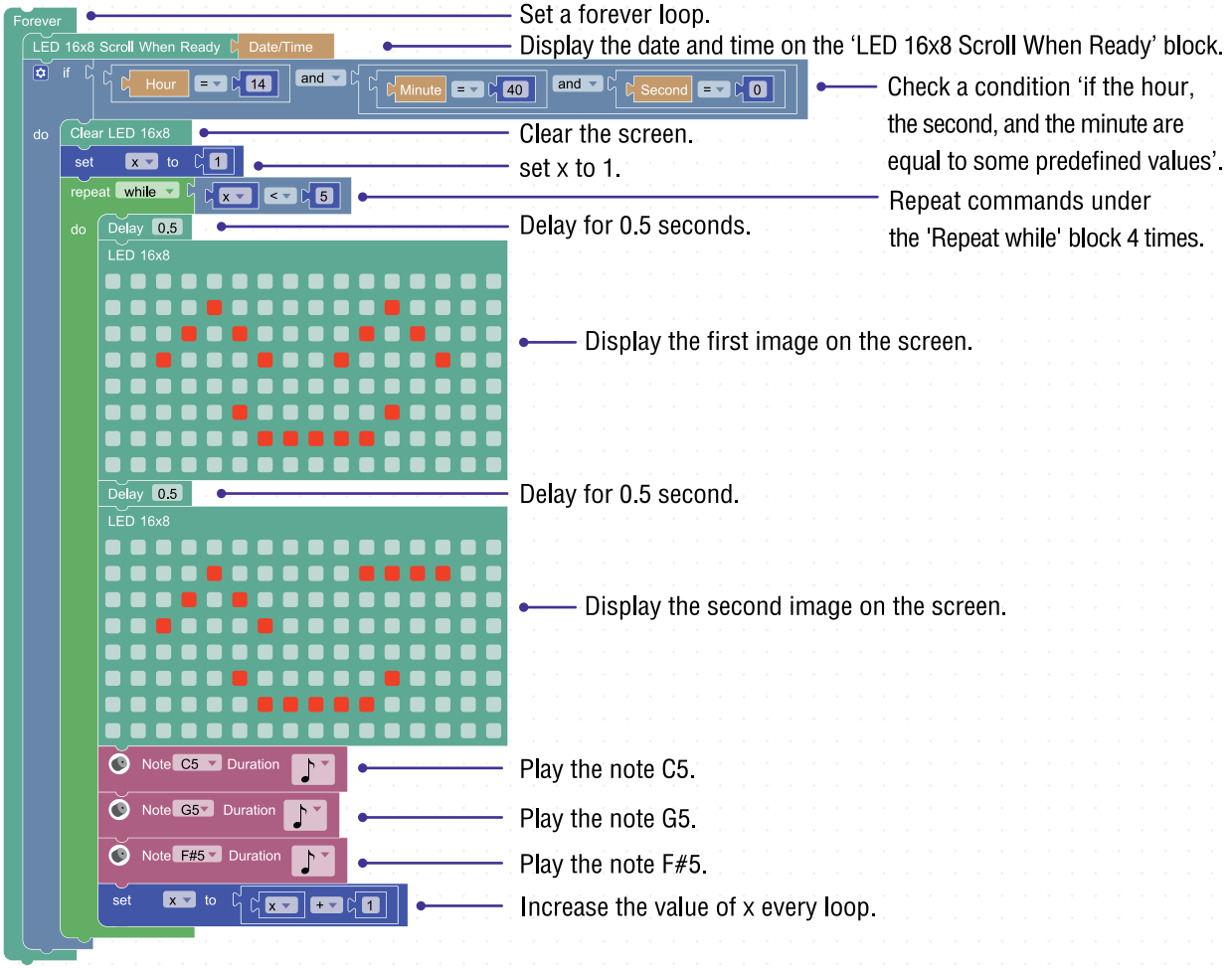
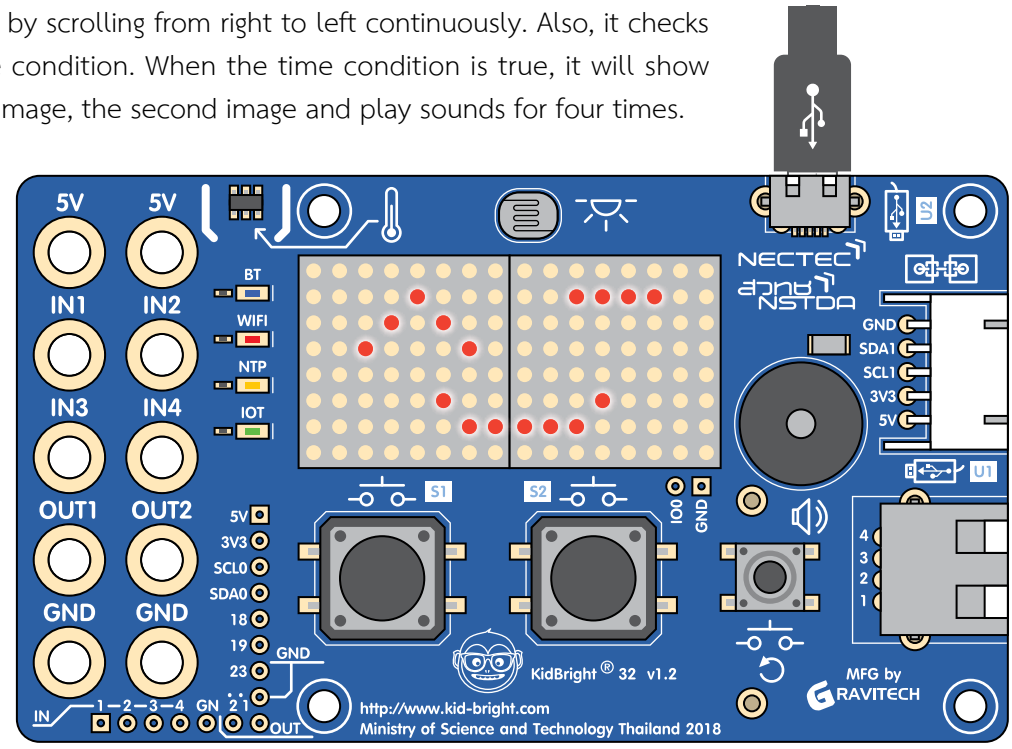


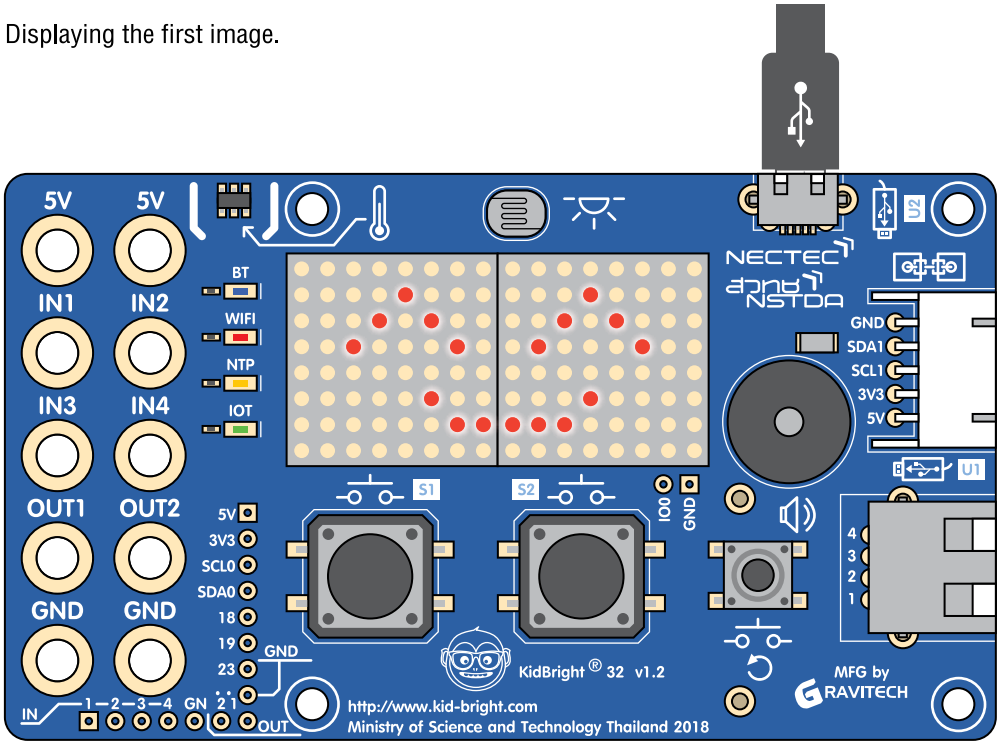
Figure 7.11 Alarm clock program.

Result

The alarm clock can be set by modifying values (hour, minute, and second) that can be used in the comparison step of the program. In this example, the clock is set at 14:40:00 The program displays the time by scrolling from right to left continuously. Also, it checks the time condition. When the time condition is true, it will show the first image, the second image and play sounds for four times.



Displaying the first image.



Displaying the second image.

Figure 7.12 Result of the alarm clock program.

Activity 7.7

This is an example of a program that turns a light on and off according to a time schedule.
Connect a light to the USB port, as shown in Figure 7.13, and write a program as shown in Figure 7.14.

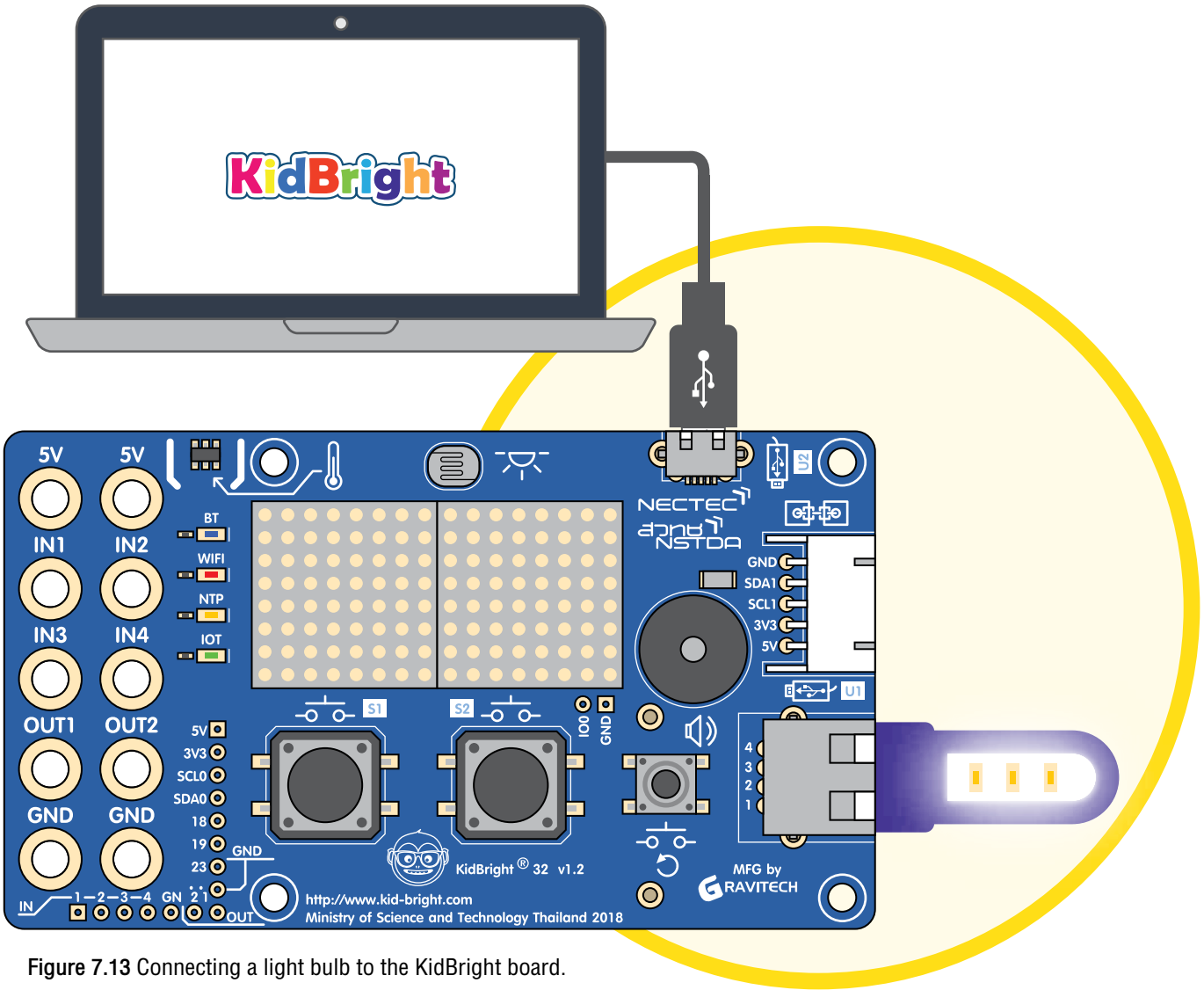


Figure 7.13 Connecting a light bulb to the KidBright board.

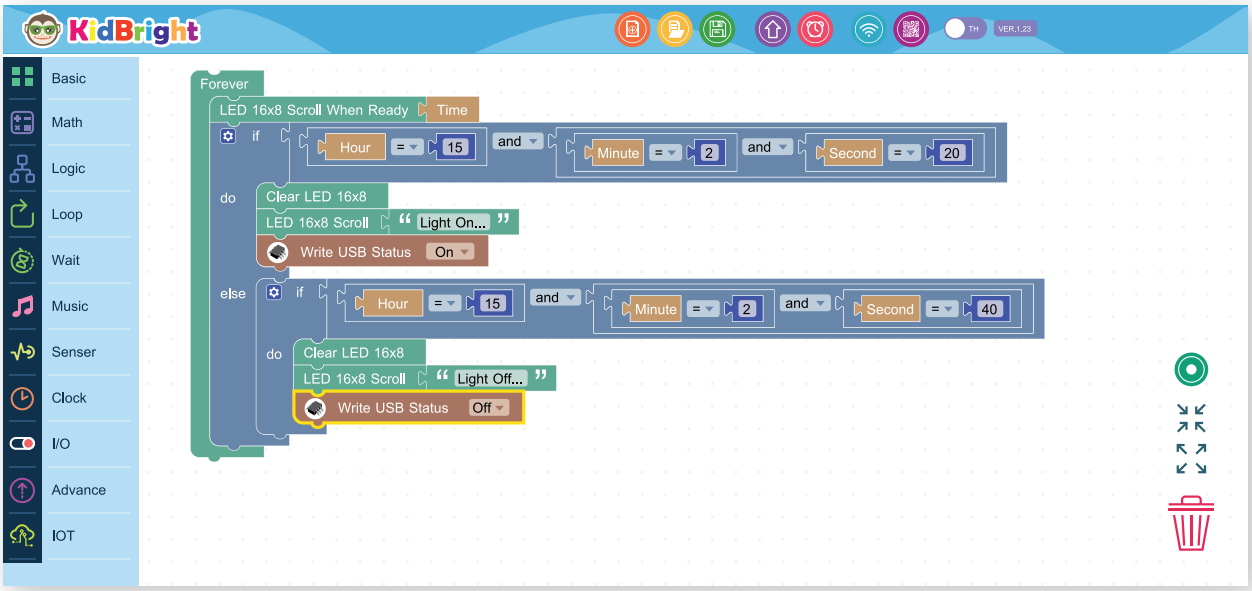


Figure 7.14 Program to control a light bulb.

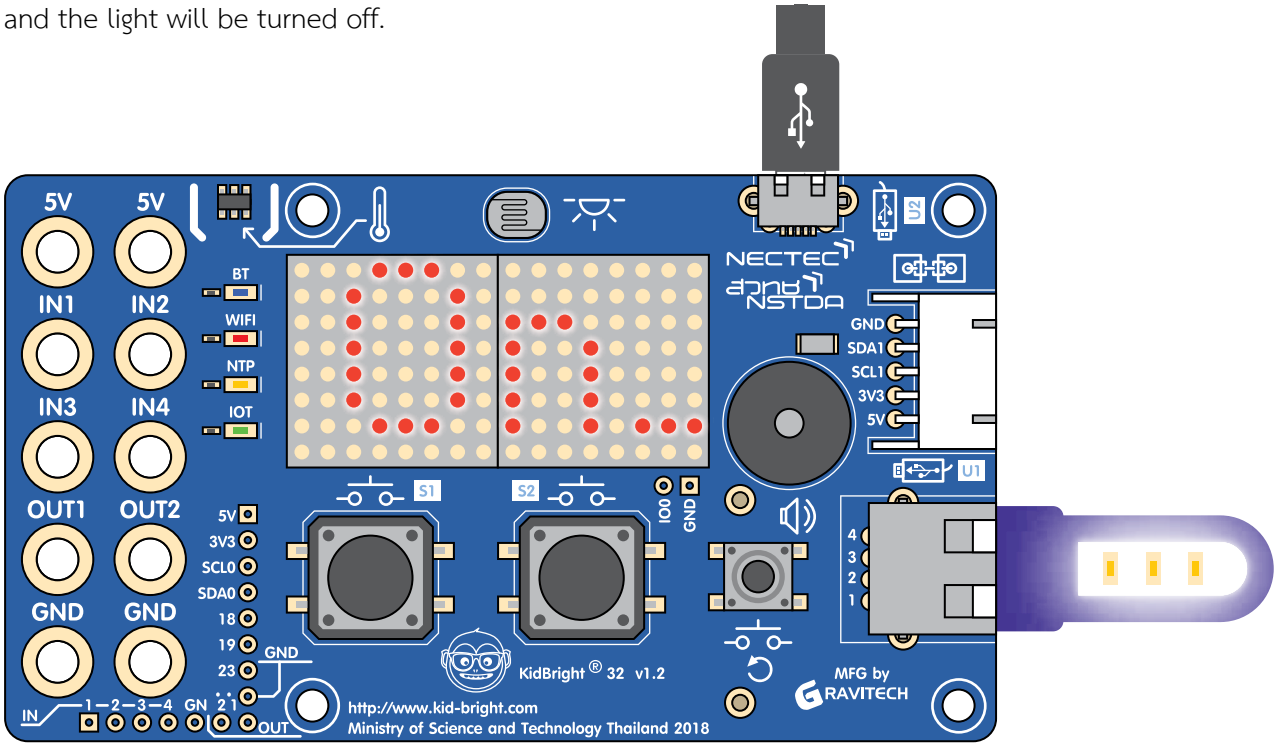
Description of program in Figure 7.14.

The program for controlling the light bulb has the following details.

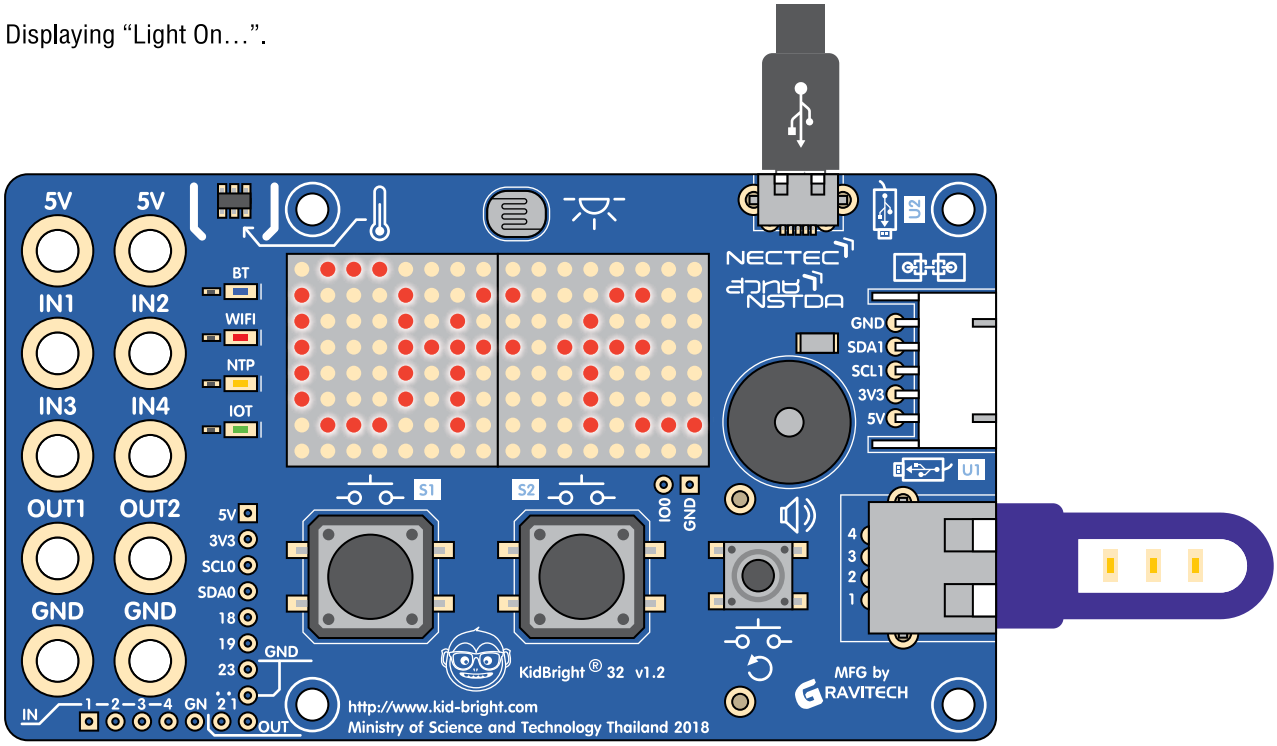
1. Set a forever loop.
2. Display the time by scrolling from right to left.
3. Check the following condition: Is it true that “The hour is equal to 15, the minute is equal to 2 and the second is equal to 20” ?
   3.1 If the condition is true, the board will clear the screen.
   3.2 The text “Light On...” will be shown on the screen.
   3.3 The board will turn on the power to the USB port so that the light is on.
4. Check the following condition: Is it true that “The hour is equal to 15, the minute is equal to 2 and the second is equal to 40” ?
   4.1 If the condition is true, the board will clear the screen.
   4.2 The text “Light Off...” will be shown on the screen.
   4.3 The board will turn off the power to the USB port so that the light is off.
5. Go to step 2.

Result

The time shows on the screen and runs from the right-hand side to the left-hand side. The time condition is repeatedly checked. When it is 15:02:20, the text “Light On...” will be displayed on the screen and the light will be turned on. When it is 15:02:40, the text “Light Off...” will be displayed on the screen and the light will be turned off.



Displaying “Light On…”.



Displaying “Light Off…”.

Figure 7.15 Displaying the results of the light-control program.

Activity 7.8

Write a timer program.

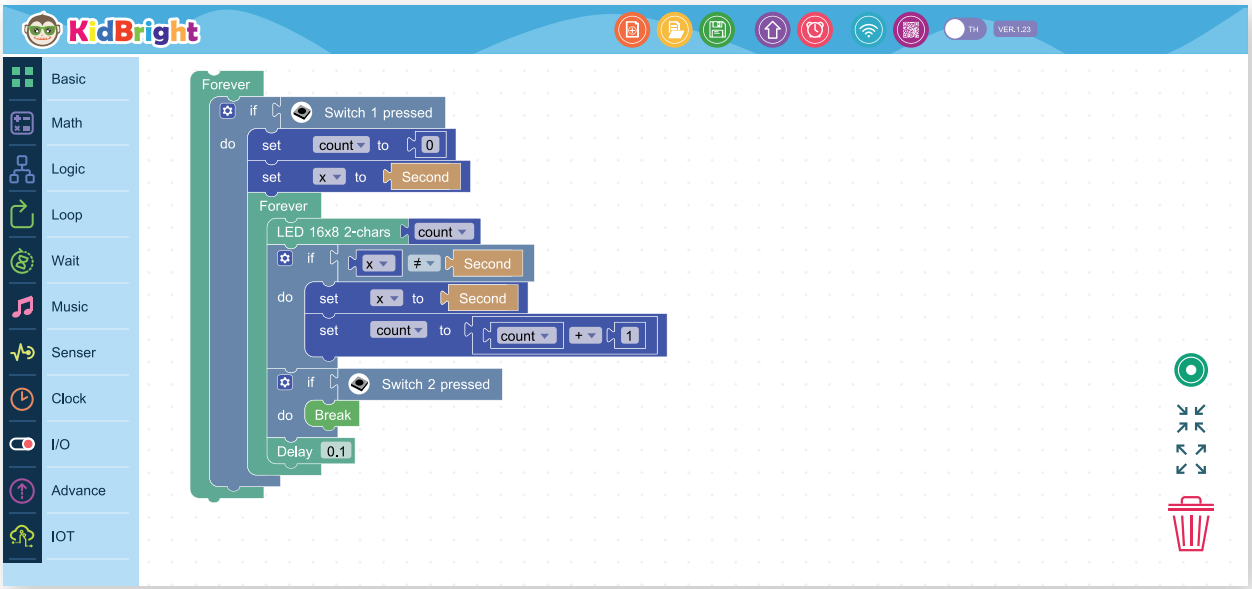


Figure 7.16 Timer program.

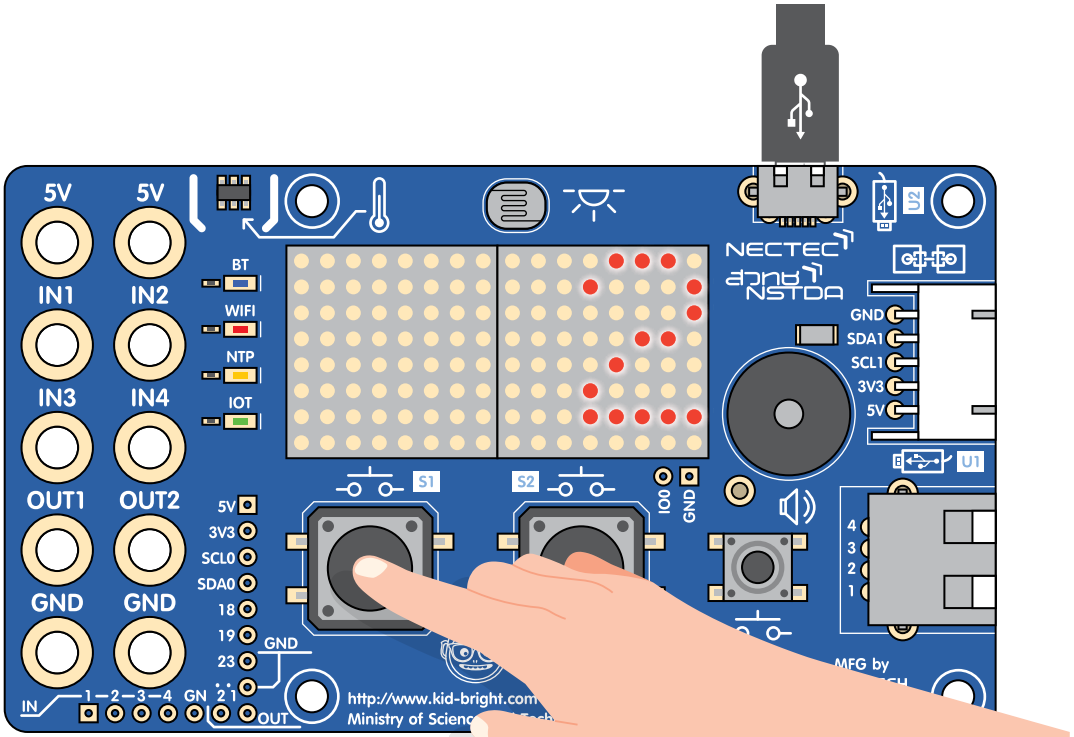
Description of program in Figure 7.16.

The timer program has the following details.

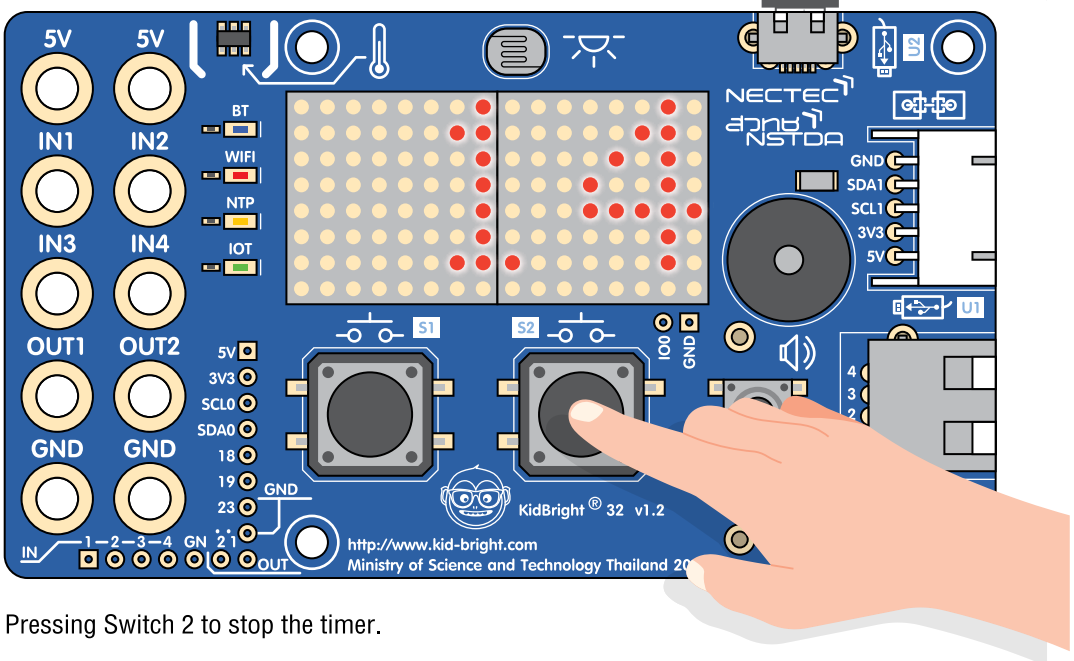
1. Set a forever loop.
2. Check the following condition: Is it true that "Switch 1 is pressed" ?
   - 2.1 If the condition is true, the value of the variable **count** is assigned to be 0.
   - 2.2 Restore the second in the variable x.
   - 2.3 Set another forever loop.
   - 2.4 Display the **count** value on the 'LED 16x8 2-chars' block.
   - 2.5 Check the following condition: Is it true that "The value of x is equal to the current second"?
     - 2.5.1 If the condition is true, restore the second in the variable x.
     - 2.5.2 Increase the **count** value by 1.
   - 2.6 Check the following condition: Is it true that "Switch 2 is pressed"?
     - 2.6.1 If the condition is true, exit the loop in step 2.3.
   - 2.7 Delay for 0.1 second.
3. Go to step 2.

Result

When Switch 1 is pressed, the timer is activated and the time in seconds is displayed. The variable x is used to keep the time. If x is not equal to the current seconds value, the value of the variable **count** is increased and is shown on the screen. The timer stops when Switch 2 is pressed.



Pressing Switch 1 to start the timer.



Pressing Switch 2 to stop the timer.

Figure 7.17 Results of the timer program.

Activity 7.9

Write a program that can detect low light intensity.

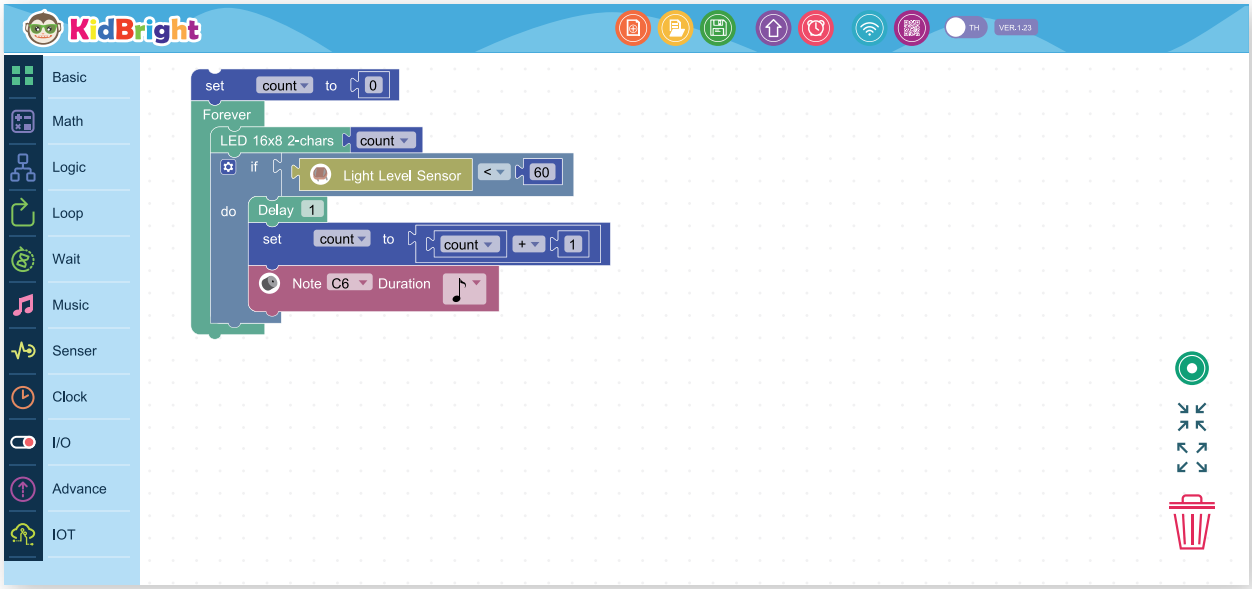


Figure 7.18 Program for detecting low light intensity.

Description of program in Figure 7.18.

The program for detecting low light intensity has the following details.

1. Set the **count** value to be 0 in order to detect a time duration during the low light intensity condition.
2. Set a forever loop.
3. Display the **count** value on the screen.
4. Check the following condition: Is it true that "the value obtained from the light sensor is less than 60"?
   4.1 If the condition is true, delay for 1 second.
   4.2 Increase the value of **count** by 1.
   4.3 Play the note C6.
5. Go to step 3.

Result

If the value read from the light sensor is less than 60 (i.e., in the low light intensity condition), the timer will start, and the sound will be played through a buzzer. Also, the screen shows time duration (in seconds) in the low light intensity condition.

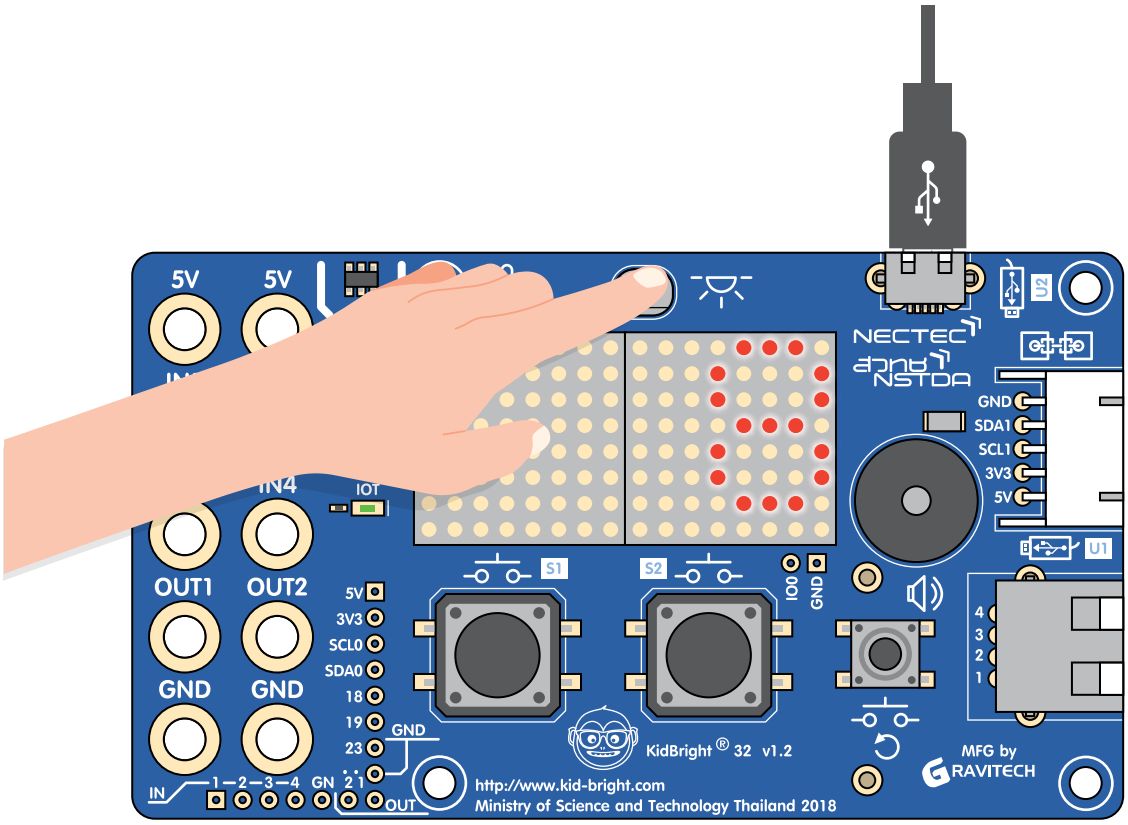


Figure 7.19 Displaying the duration during the low light intensity.

Activity 7.10

Write a program that can detect a high temperature.

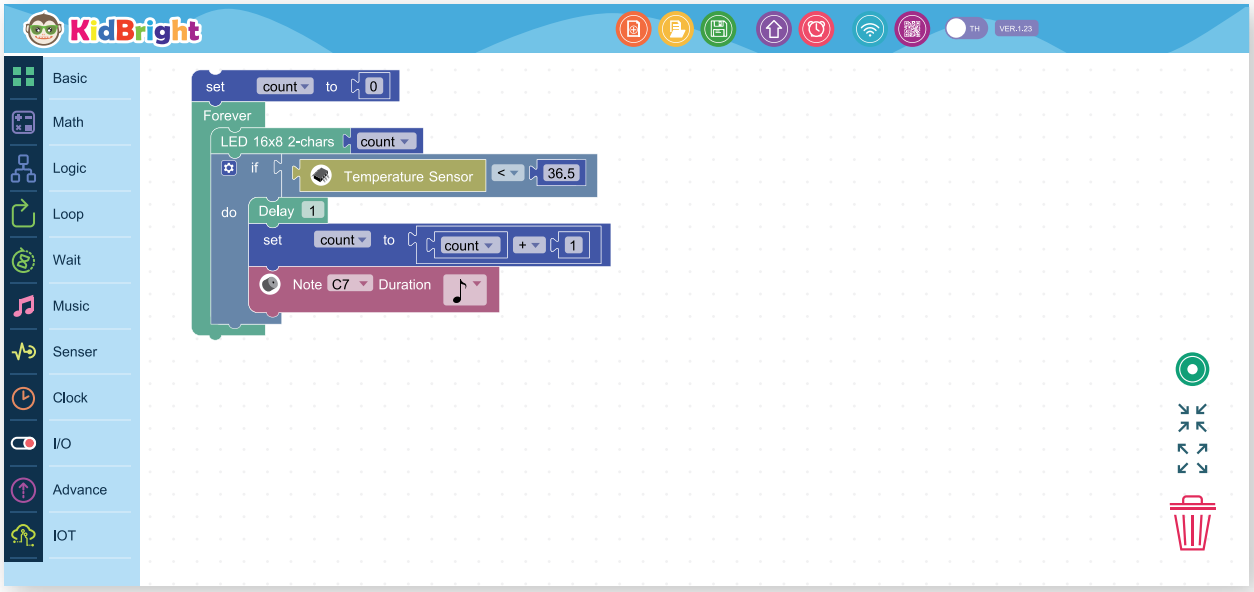


Figure 7.20 Program for detecting high temperature.

Description of program in Figure 7.20.

A program that can detect a high temperature has the following details.

1. Set the **count** to 0 in order to detect a time duration during the high temperature condition.
2. Set a forever loop.
3. Display the **count** on the screen.
4. Check the following condition: Is it true that "the value obtained from the temperature sensor is greater than 36.5"?
   4.1 If the condition is true, delay for 1 second.
   4.2 Increase the value of **count** by 1.
   4.3 Play the note C7.
5. Go to step 3.

Result

If the value read from the temperature sensor is greater than 36.5 (i.e., in the high temperature condition), the timer will start and the sound will be played through a buzzer. Also, the screen shows time duration (in seconds) in the high temperature condition.

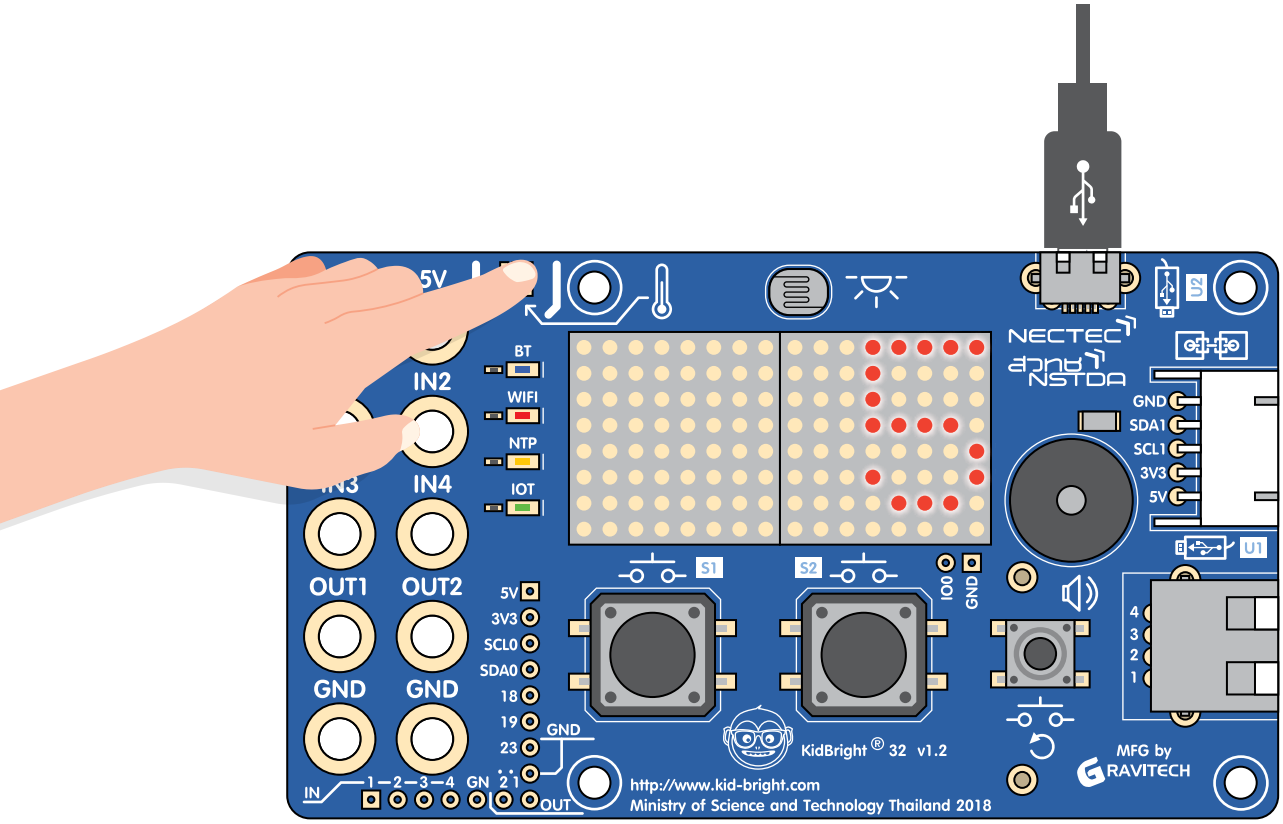


Figure 7.21 Displaying the time duration during the high temperature.

Multitasking programming

The processing unit of early computers had limited computing power. Multiple tasks had to be programmed and done in sequence. The example in Figure 7.22 shows such a sequential order in which multiple tasks are executed one by one. Today, we have high-performance processing units that can do more than one task at a time (multitasking programming), as shown in Figure 7.23.

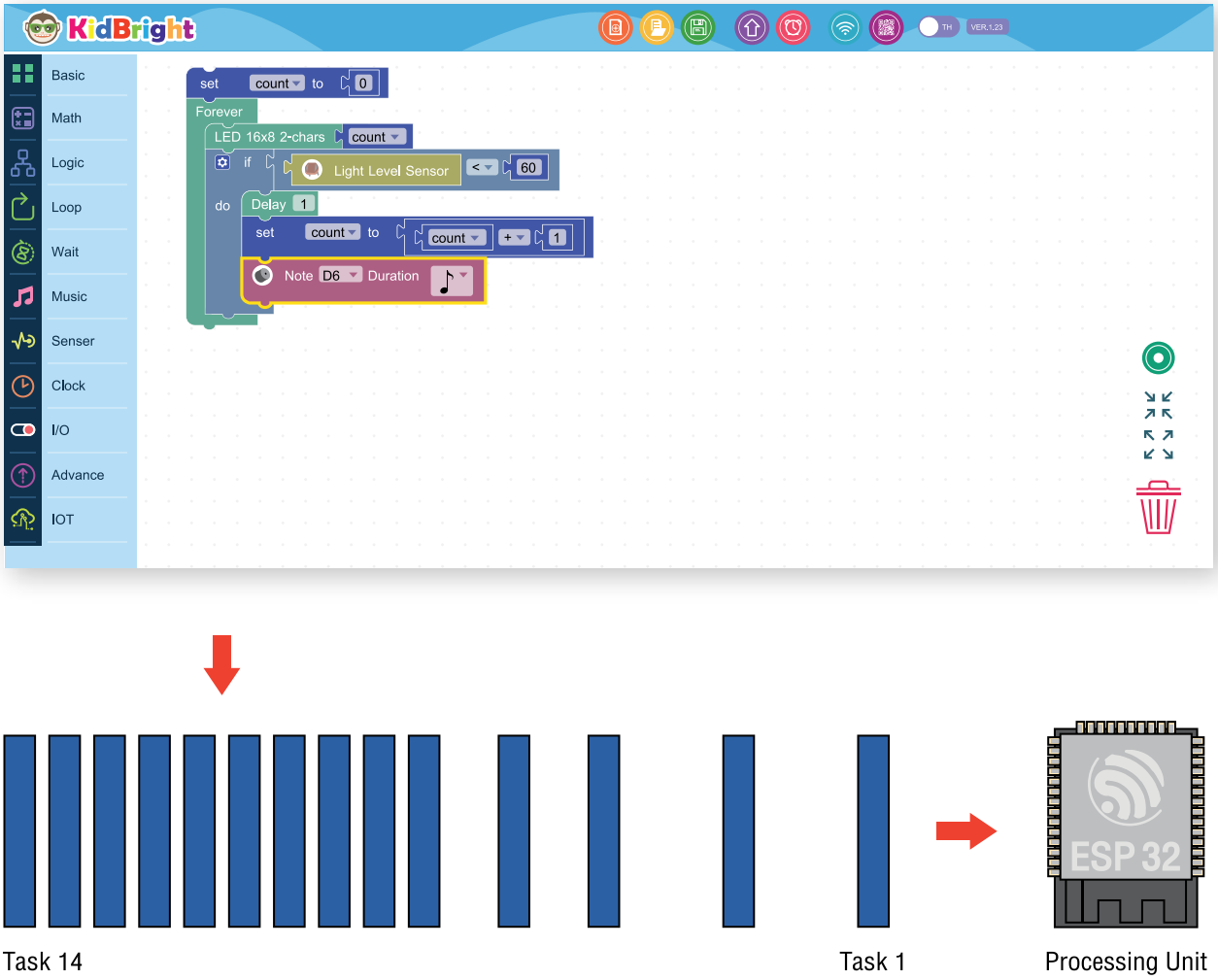


Figure 7.22 Sequential programming.

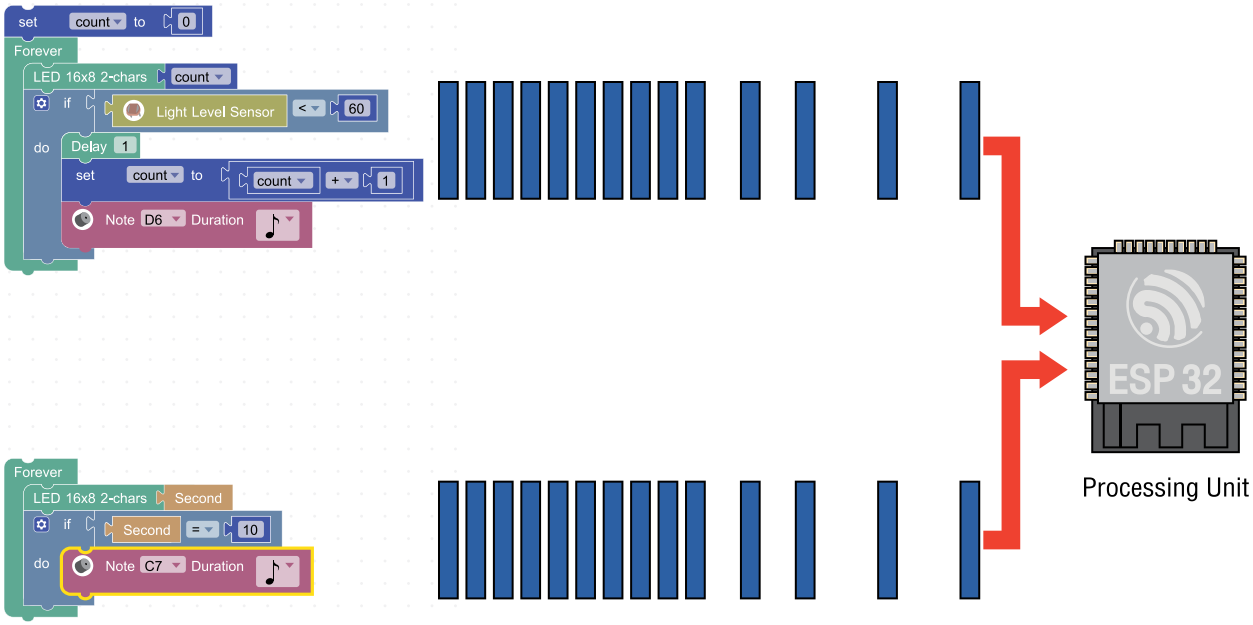


Figure 7.23 Multitasking programming.

The 'Task' block, which is on the Advance tab, can be used to write a multitasking program. The 'Task' block is shown in Figure 7.24, and students can assign a task title to the 'Task' block.

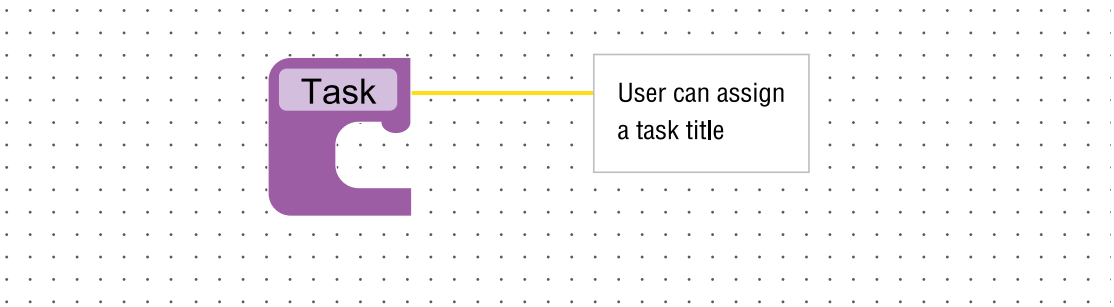


Figure 7.24 'Task' block on the Advance tab.

Activity 7.11

Write a multitasking program.

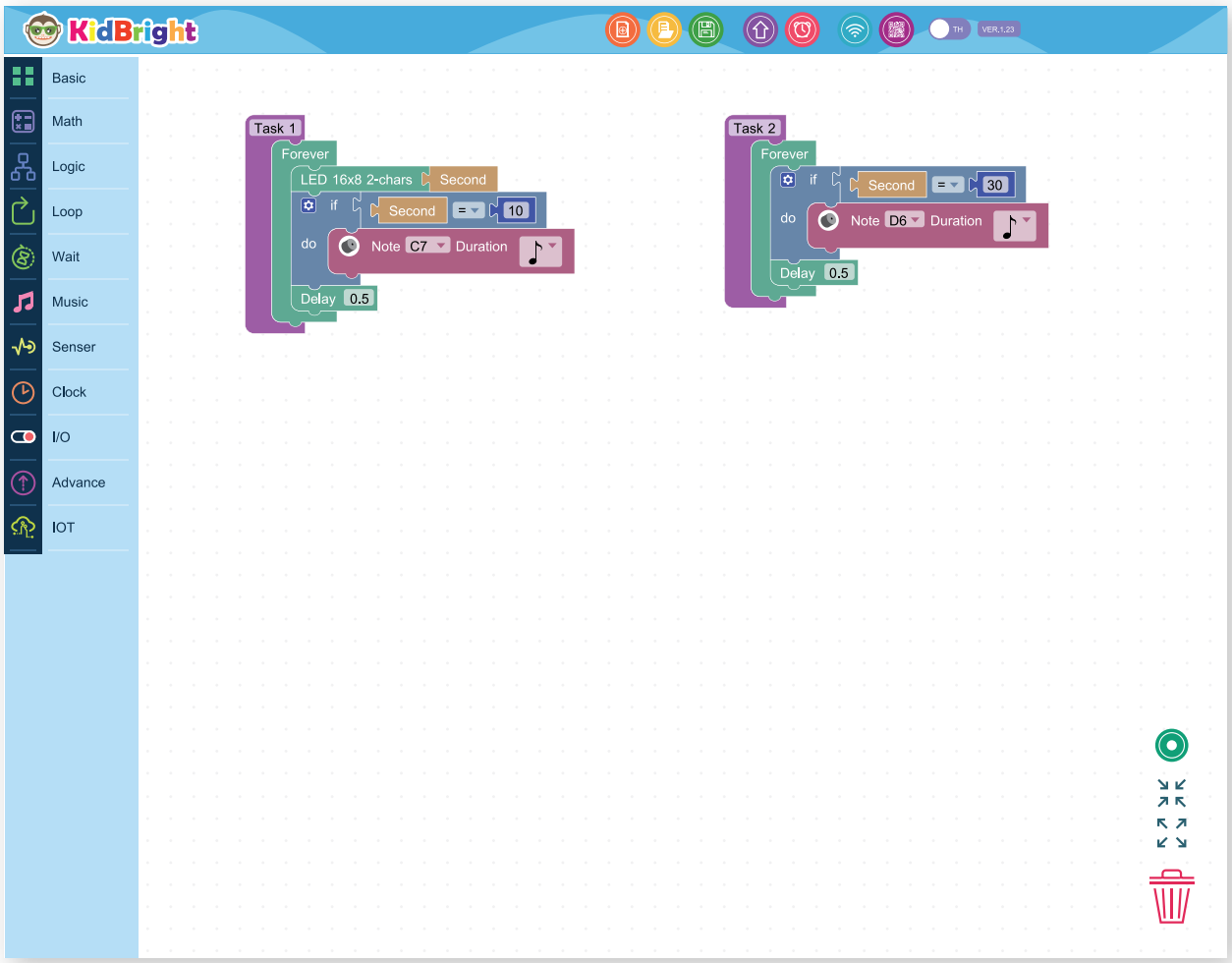


Figure 7.25 Multitasking program.

Description of program in Figure 7.25.

There are 2 tasks done in parallel as follows:

Task 1

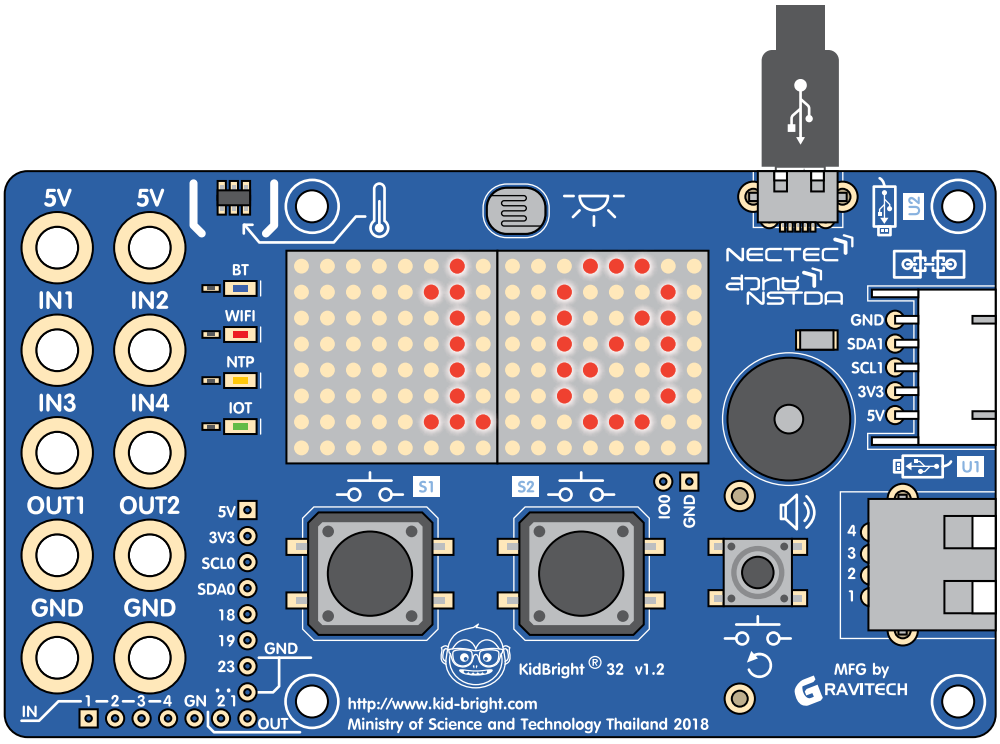
1. Set a forever loop.
2. Display the seconds on the 'LED 16x8 2-chars' block.
3. Check the following condition: Is it true that the seconds value is equal to 10"?
   3.1 If the condition is true, play the note C7.
   3.2 Delay for 0.5 second.
4. Go to step 2.

Task 2

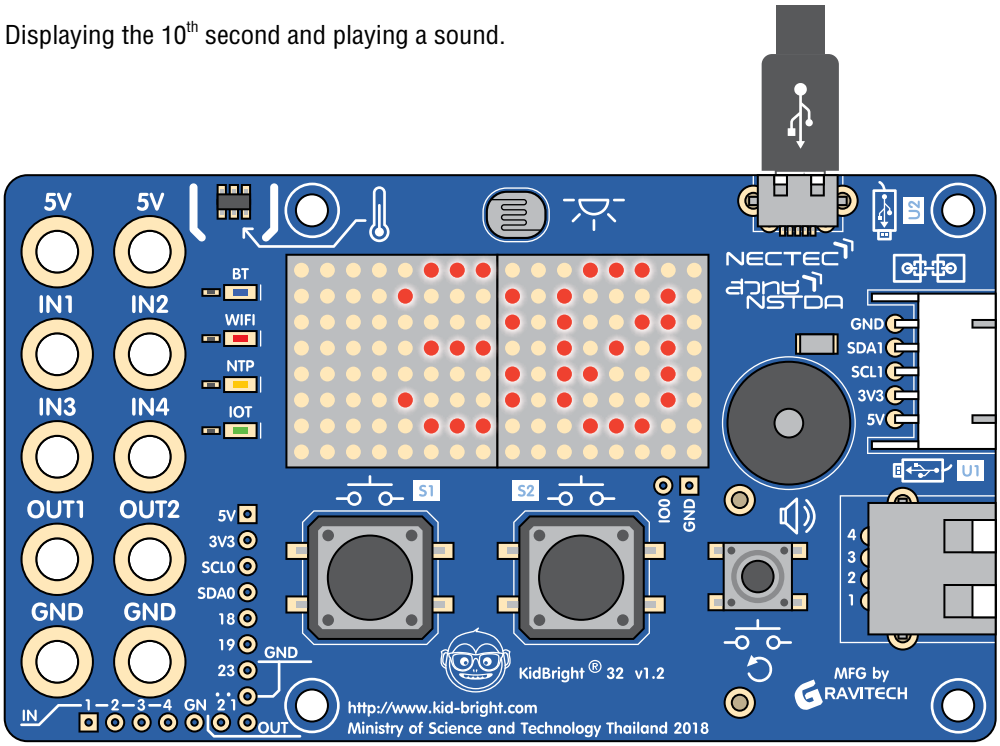
1. Set a forever loop.
2. Check the following condition: Is it true that the seconds value is equal to 30"?
   2.1 If the condition is true, play the note D6.
   2.2 Delay for 0.5 second.
3. Go to step 2.

Result

The seconds value is displayed on the screen. If it is equal to 10, the sound C7 will be played through the buzzer. Furthermore, if the seconds value is equal to 30, the sound D6 will be played.



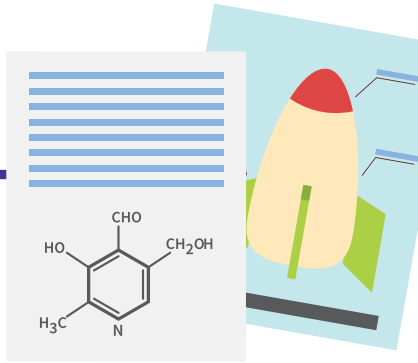
Displaying the 10th second and playing a sound.



Displaying the 30th second and playing a sound.

Figure 7.26 Results of the multitasking program.

Exercise

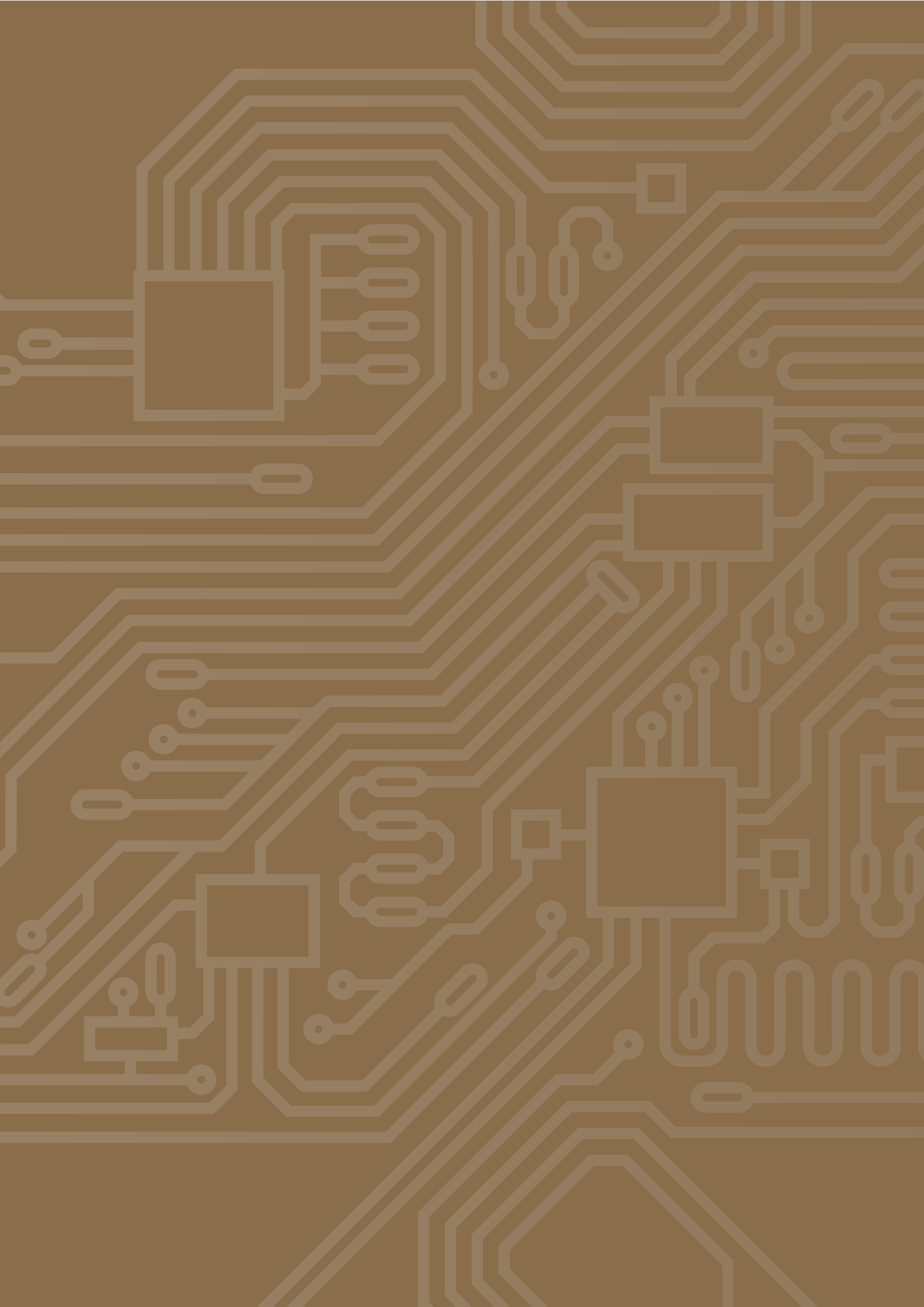


1. Explain functions of the multitasking.

2. Write a program that turns on a light at 18:00.

Appendix

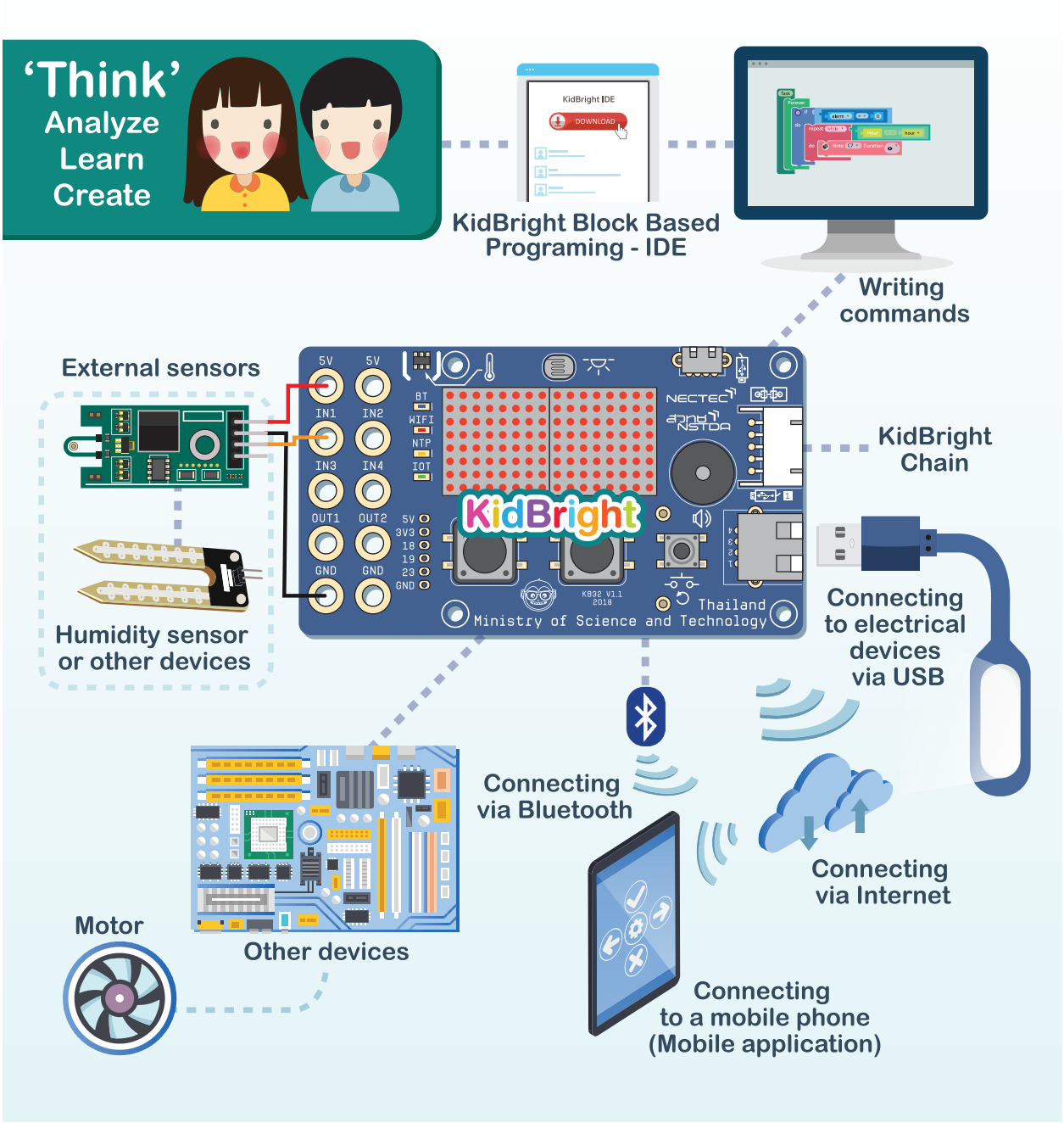
Overview of KidBright



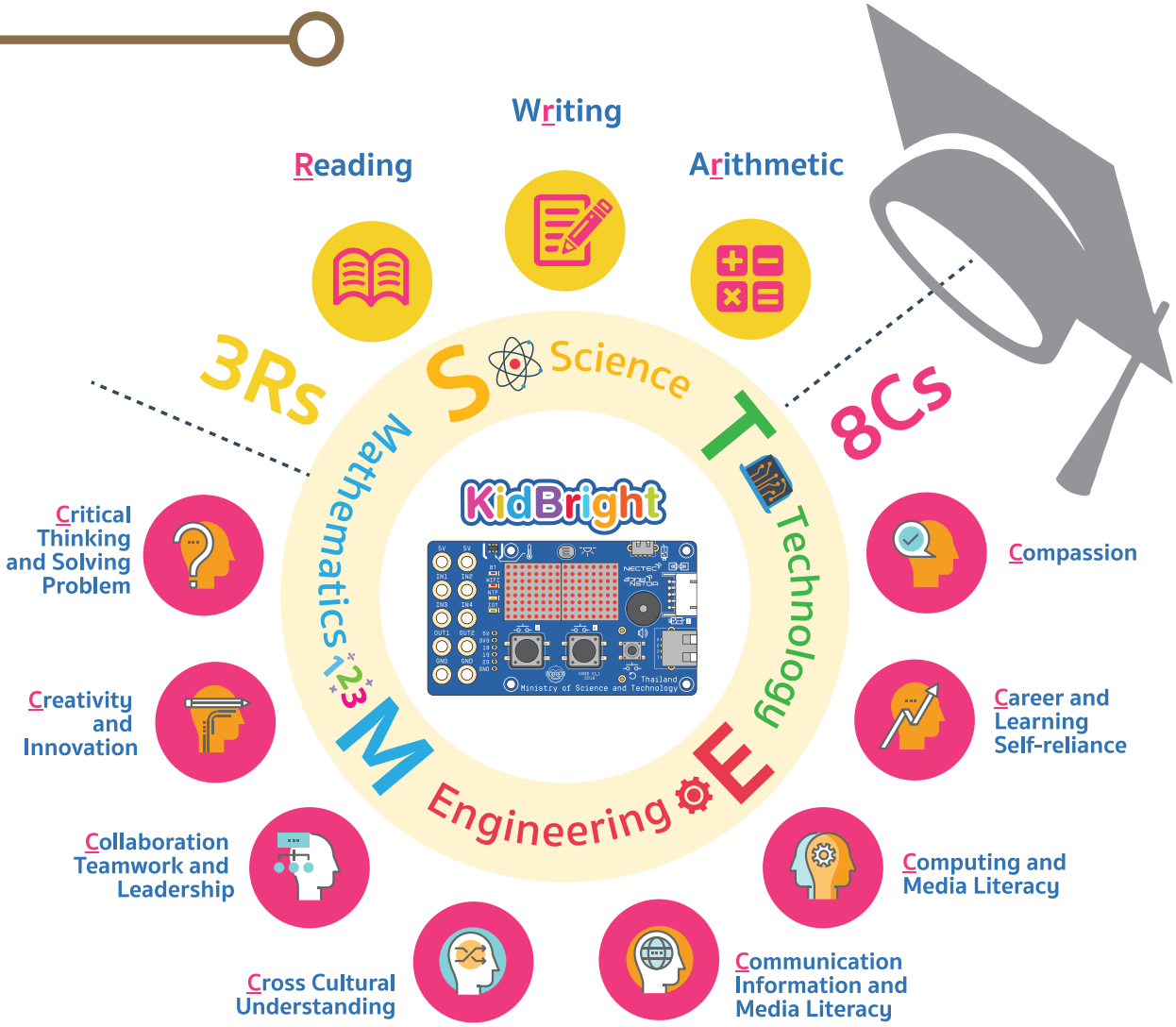
KidBright Overview

KidBright is an embedded board or a tiny computer that has a LED display and sensors. It is able to connect to other devices via WiFi, or Bluetooth. It can work with offline and online modes, i.e., Internet of Things (IoTs).

You can create a watering machine, an automatic lamp, a waste segregation machine, a thief detector at home and even novel more schematics. The possibilities are endless.



Learner can design and create a set of commands using Block Based programming that is easy and fun. Furthermore, it promotes thinking processes, i.e., systems thinking, analytical thinking and creative thinking. KidBright can thus be a part of science, technology, engineering and mathematics (STEM) education that is promoted in many countries worldwide.



Following up our information at

KidBright

www.kid-bright.org

E-mail : kidbright@nectec.or.th

KidBright installation

Step 1

1

Download KidBright IDE
(64 bits, New UI, Plugins beta version)
and FDTI Driver from.

www.kid-bright.org

Step 2

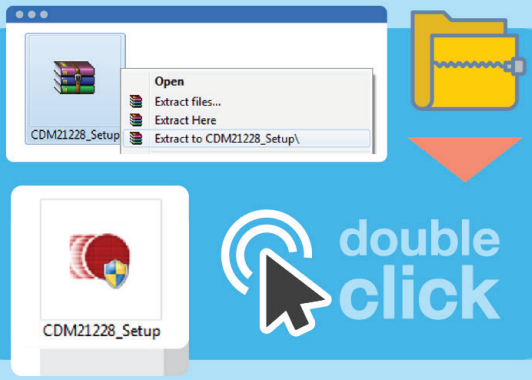
2

Unzip KidBright-installer-windows
and double click KidBright-installer
to install.

Step 3

3

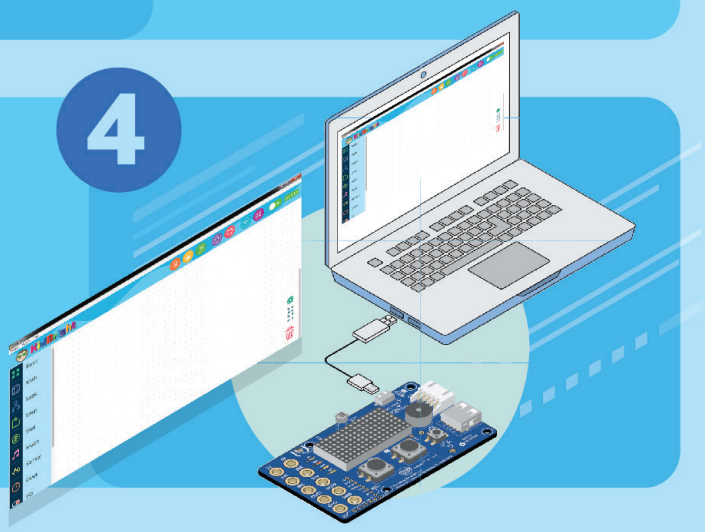
Unzip CDM21228_Setup
and double click CDM21228_Setup
to install.



Step 4

4

Connect KidBright
to computer using USB
cable.

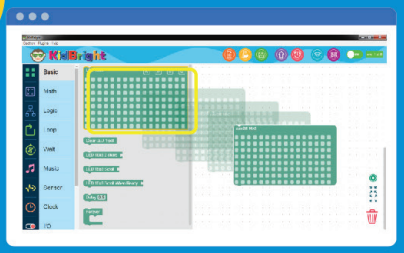


Coding with KidBright

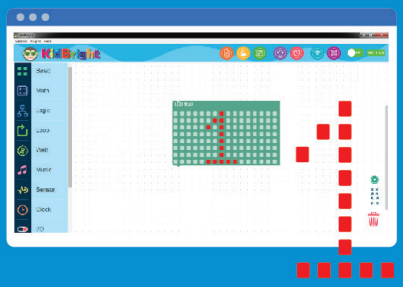
1

Step 1

Drag the LED 16x8
from "Basic" tap
to the empty space



2



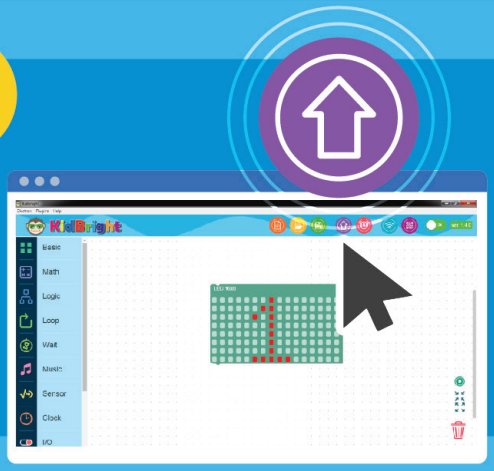
Step 2

Click dots to red for enabling
LED display. All red dots form '1'
on LED screen.

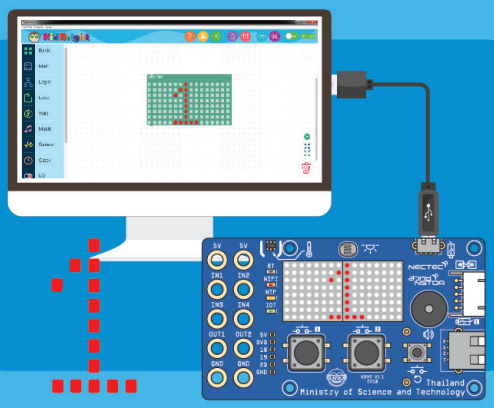
3

Step 3

Click "Build program."



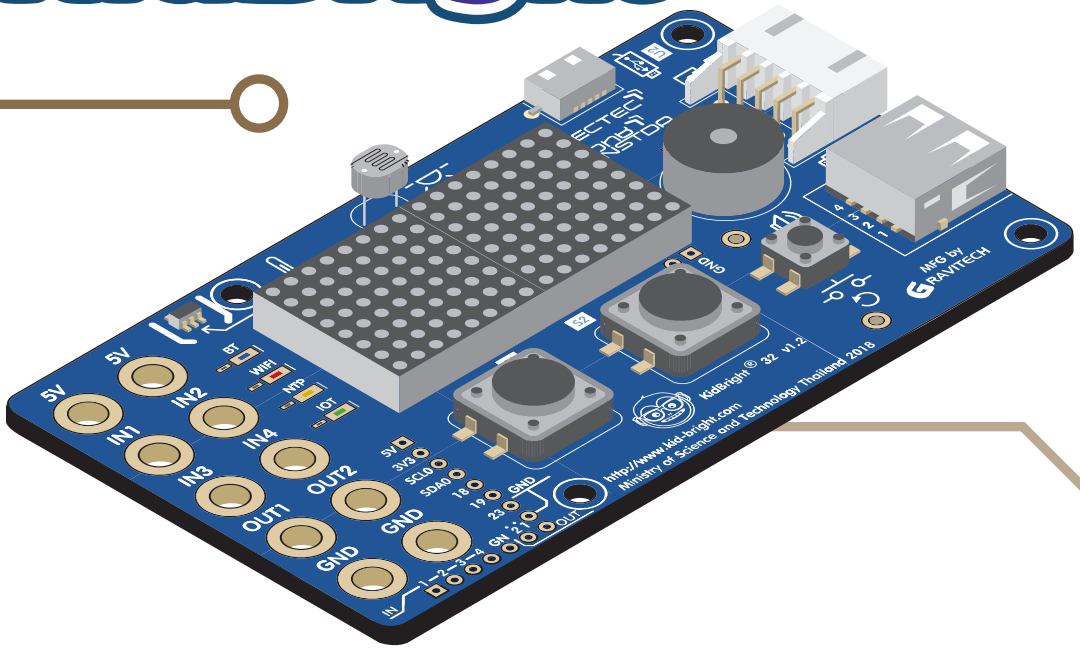
4



Step 4

KidBright IDE will convert commands
to binary code and send to the board.
The KidBright displays '1' on LED.

KidBright


5V
5V
IN2
IN1
IN4
IN3
OUT2
OUT1
GND
GND
BT
WiFi
MP
IOT
5V
3V3
SCL0
SDA0
18
19
23
GND
GND
MFG by
GRAVITECH
KidBright® 32 v1.2
http://www.kid-bright.com
Ministry of Science and Technology Thailand 2018

Advisors

Professor Dr.Pairash Thajchayapong

Dr.Thaweesak Koanantakool

Authors

Chirasak Suwanno

Rawat Jaisutti

Chaiwut Sreesawatt

Kittikhun Sa-ard

Supista Matra

Suphanut Thanyaboon

Pinyo Yonthanthum

Editors

Associate Professor Dr.Santi Wijakkanalan

Chittrakorn Pantarach

Assistant Professor Dr.Chayakarn Keereerat

Dr.Piyanee Chitcharoen

Dr.Pinyo Yonthanthum

Dr.Kitisak Kerdto

Editors from NECTEC

Dr.Sarun Sumriddetchkajorn

Dr.Kalaya Udomvitid

Dr.Saowaluck Kaewkamnerd

Dr.Apichart Intarapanich

Dr.Anuchit Leelayuttho

Peeranan Kanjanasrisuntorn

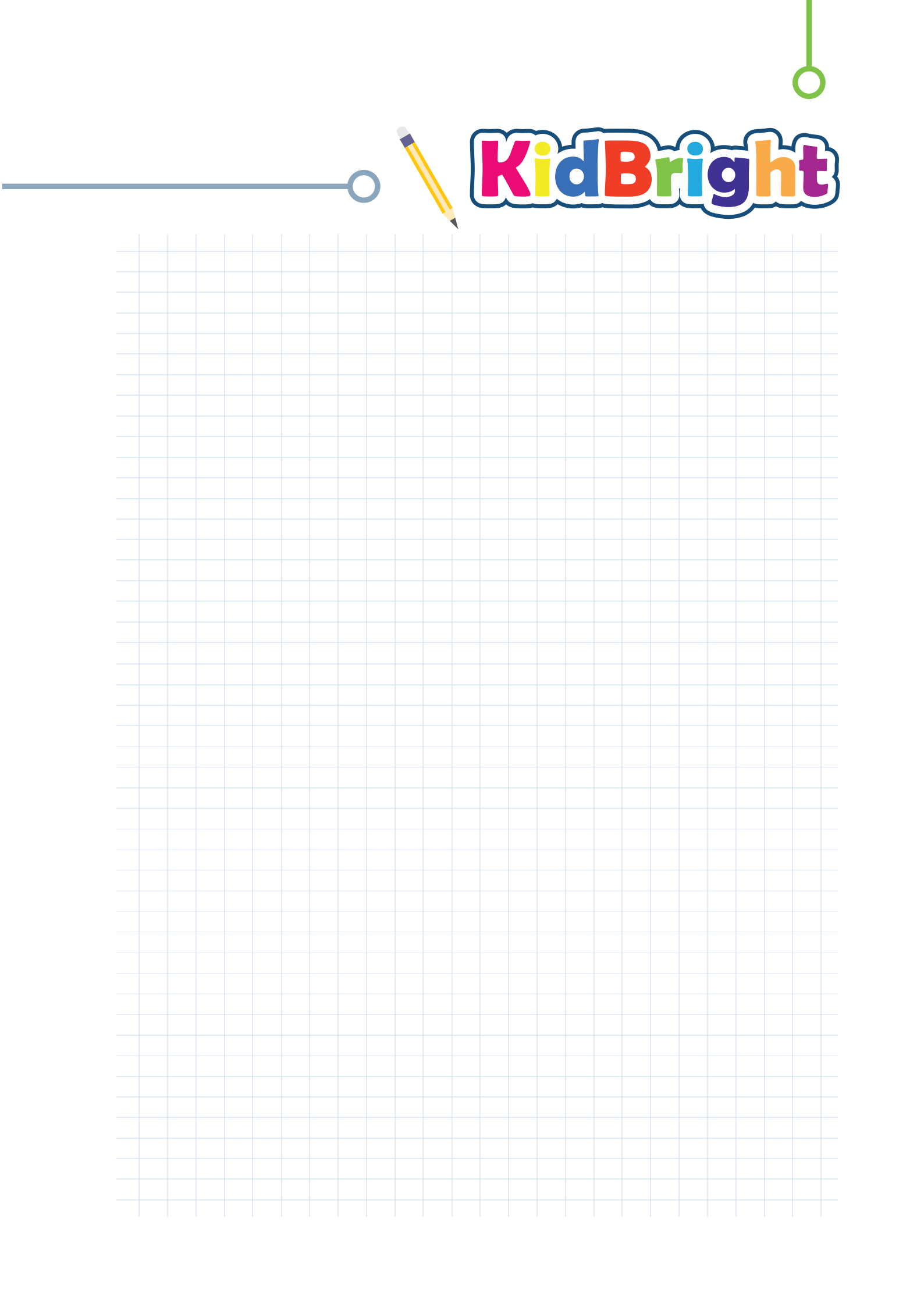
Supara Phantiya

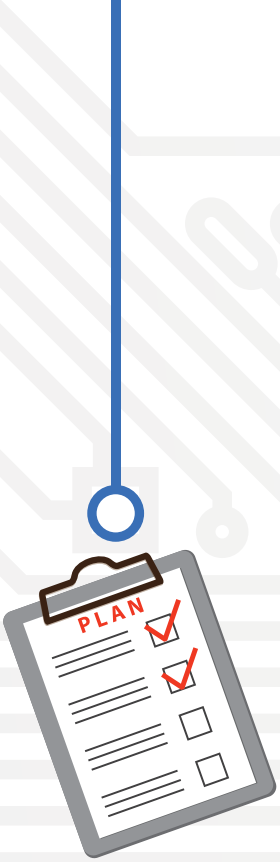
Kantavee Pansitha

Proofreaders

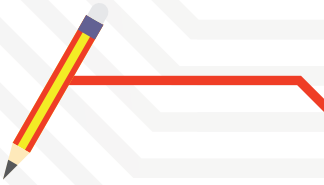
Philip James Shaw

Surapol Tan-a-ram


KidBright



KidBright



9 786161 205539

200.-