PROGRAMMING EDUCATION ROBOT

LEARNING LAB

#1276

274 PCS

4+

INVENTING CAN BE LEARNED

20 EXPERIMENTS INCLUDED

INVENTING CAN BE LEARNED

Gigo Learning Lab's complete series includes individual packages and school sets. The special features of Gigo's Learning Lab are as follows:

1. Using Gigo's building block construction-based curriculum, every class has a ready-to-assemble model, and includes time designated to promote individual creativity.

2. Boots thinking outside-the-box of the traditional educational framework by learning innovation through play!

3. We are all innately good at something, so we should take into account both individual development and the ability to work as part of a team.

4. Course levels are designed from elementary to challenging, combining a life sciences-based curriculum with applications from daily life.

5. Experiment using Gigo's building blocks, which can be used over and over again, saving both time and effort.

We hope that kids can enthusiastically learn scientific knowledge through fun hands-on experience, developing their problem-solving abilities, as well as a positive attitude towards science. Our mission is to help children apply their newfound knowledge to daily life, furthering their innovational skills and abilities.

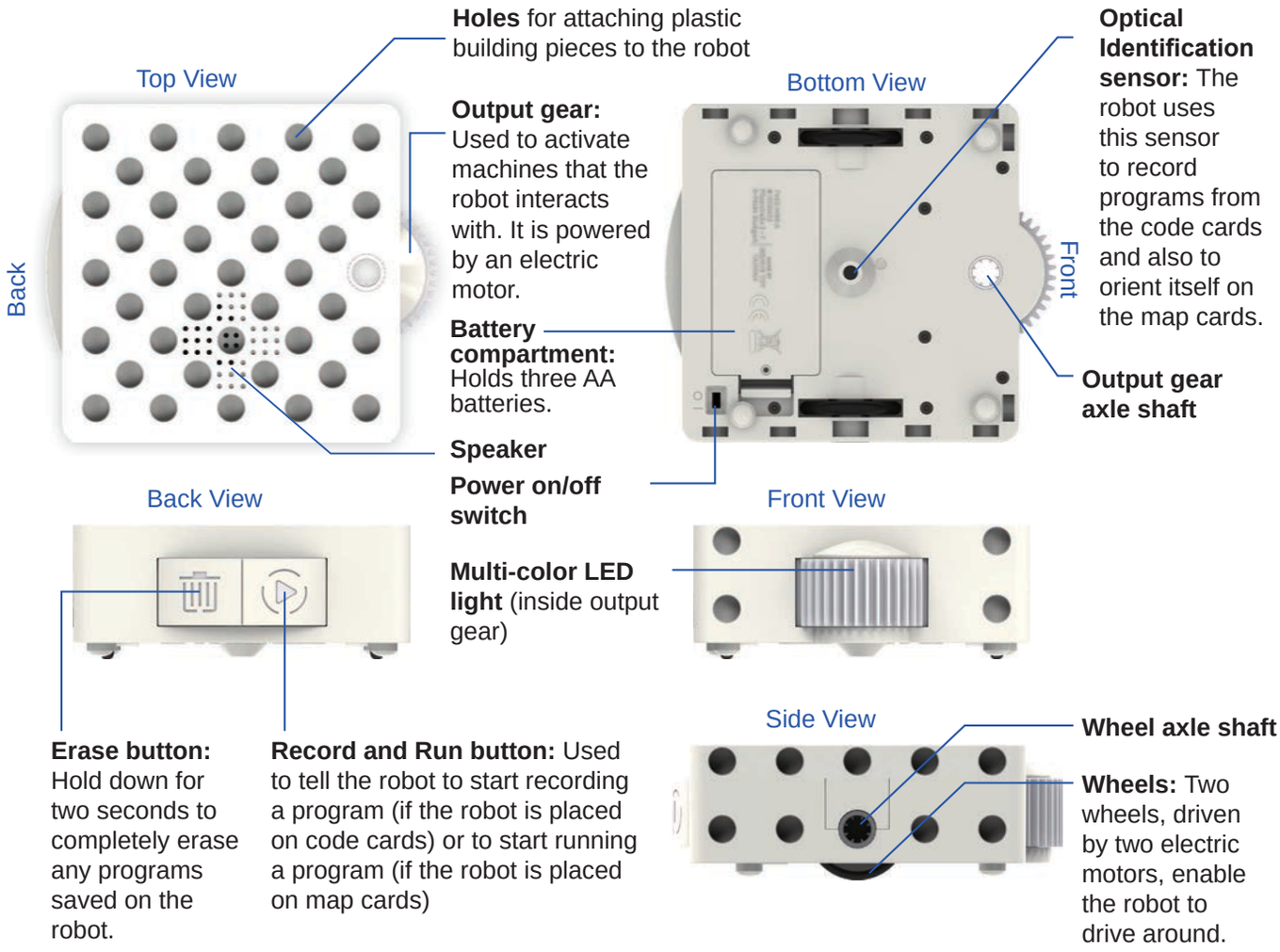
Index

Getting Started

Welcome to Programming Education Robot! First, let's take a look at the main parts of this kit: the robotic base unit, code cards and map cards.

Robotic Base Unit

This is the base for all the robots you can build with this kit. Step-by-step instructions starting on page 18 show you how to assemble the plastic building pieces in the kit onto the robotic base unit, or into other models that can be used alongside the robots you construct. The robotic base unit is packed with cool functionality! Here's an overview of all of its features:



Holes for attaching plastic building pieces to the robot

Output gear: Used to activate machines that the robot interacts with. It is powered by an electric motor.

Battery compartment: Holds three AA batteries.

Speaker

Power on/off switch

Multi-color LED light (inside output gear)

Optical Identification sensor: The robot uses this sensor to record programs from the code cards and also to orient itself on the map cards.

Output gear axle shaft

Erase button: Hold down for two seconds to completely erase any programs saved on the robot.

Record and Run button: Used to tell the robot to start recording a program (if the robot is placed on code cards) or to start running a program (if the robot is placed on map cards)

Wheel axle shaft

Wheels: Two wheels, driven by two electric motors, enable the robot to drive around.

Normal Mode vs. Math Mode

The robot has two modes. Out of the box, the robot is in **normal mode** by default. Lessons 1–15 use normal mode only. The robot has to be set to **math mode** to do the math lessons at the end of the manual. Learn how to use math mode starting on page 63.

Overview of Normal Mode Operation

In normal mode, when you slide the robot's power switch to the on position, the robot stands by for recording. You can then have the robot record a program. After the robot successfully records a program, the robot stands by to begin execution of the program. Place the robot on the Start map card, and the program will begin to run. When the program ends, the robot stands by to either run the same program again or record a new program.

Code Cards

To program the robotic base unit, you don't need a computer or a tablet — all you need are the **code cards** and the **code card frames**! There are 61 different code cards. The kit includes multiple copies of some of the cards. There are 108 double-sided code cards, for a total of 216 sides.

You write a program by laying out a sequence of code cards in the frames. Then, the robot drives over the code cards one by one. While it does this, the optical identification sensor on the bottom of the robot scans a small pattern of dots that you can barely see printed on the cards. The robot's microprocessor is preprogrammed to translate this pattern into instructions it can follow.

Every program always starts with a **Start** code card.

Every program always ends with an **End** code card.

There are also cards that make the robot **move**.

There are cards that make the robot's **output gear turn.**

There are cards that tell the robot to make **sounds.**

There are cards that tell the robot to **light up** in a certain way or with a specific color.

There are **number** cards which **repeat** the card immediately before it a number of times.

There are simple loop cards, You will learn about all of these later in the manual.

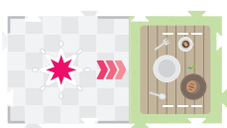
How to Use Simple Loops

There are two sets of simple loop cards in this kit: green and red. This means you can use up to two loops in the same program.
To set up a loop, you must always use two loop cards of the same color (either two green loop cards or two red loop cards). One loop card is placed at the start of the loop and the other is placed at the end of the loop. A number card must be placed immediately after the first loop card.
This number card indicates how many times the other code cards placed after it but before the second loop card will be executed (run).

You cannot place more than one number card after the first loop card. You cannot place a number card after the second loop card. Both of these placements will result in an error. You can nest one loop inside another.
You can experiment with simple loops in Lesson 6.

How to Use Functions

In the coding language in this kit, functions are demonstrated with the red, green, and blue function cards. These functions are always used with the base map cards. For example, the red function is performed when the robot scans the red function base map card.
You can learn how to use these functions in Lesson 8 and Lesson 12.



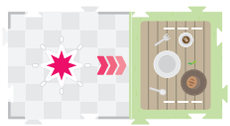
The red function card is always used with the red base map card.

You can have up to 15 code cards in a function. The **Move Forward** and **Move Backward** code cards **don't work in functions**; if you try to use them, you will get an error.
The **Turn Output Gear** and **Pause Output Gear** code cards don't work in the main program. You can use these **only in functions.**

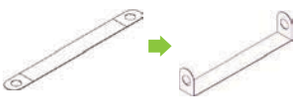
Functions are programmed with the Function Start cards: 

Getting Started

A function runs when the robot scans the **star** on the corresponding base map card, when it is **facing in the same direction** as the three arrows on the base map card, and when there is a corresponding **function program** recorded in its memory.

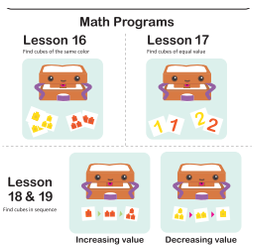


The robot must be programmed to be facing the direction of the interaction position (i.e., facing in the same direction as the three arrows). The robot can either enter the map card already facing this direction, or it can be turned with a turn code card to face this direction after entering the card.
When the robot scans the base map card, it first orients itself on the star. Then, the robot advances toward the interaction point following the three arrows. Then, the function runs. Finally, the robot backs up to the star again.
When you want to use the output gear with a model on the base map card, you need to secure the model with the **map card straps**, so it stays in place:



Math Lesson Mode

The robotic base unit can be switched into special modes to teach specific math lessons.
In these modes, the robot behaves differently than in its normal operating mode. You switch the robot into these modes by scanning the additional invisible control graphics printed on page 73.



These are just like code cards, but they are printed into the manual instead of onto separate cards. In math mode, you program the robot the same way as before, but this time with the goal of solving the stated math problem. In math mode, when the robot reaches the end of its program, it will play music and light up depending on whether the final solution was right or wrong: Harp music and multi-colored lights will play if the solution was correct. Tuba music and red-orange lights will play if the solution was incorrect.

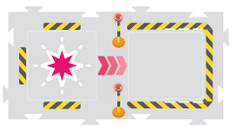
These cards represent the numbers 1 through 5 in orange and in yellow.
To complete each math lesson, write a code to solve the stated problem by moving the robot to specific numbered map cards and finally to the blue, red, or green function base map card.

Note: You cannot use function code cards or conditional code cards in math mode. The robot will not react to event cards in math mode.
To exit math mode, press and hold the Erase button for two seconds.

Map Cards

The robot always plays out (or runs) its programs on the map cards. The map cards also have invisible patterns of dots printed all over them. The robot uses the optical identification sensor to read these patterns, which tell it which map card it is on and helps it orient itself and move in the correct directions on the map cards.
For every program you write, you always lay out a grid of map cards for the robot to run its program on.

The robot always starts its program on the **Start** map card.



A few of the map cards are bigger than the others. These are called **base map cards.** You attach certain models to these cards using the **map card straps** so that the robot can interact with the models.

There are a total of 38 different map cards in the kit, including two Start cards, six base map cards, and four event map cards. The map cards are **double sided**, so there are actually just 19 separate cards, each with one map card on the front and one on the back.

The map cards have **interlocking tabs** like a jigsaw puzzle to keep them together. Please note that you either have to use the front sides or the back sides of the map cards at one time, because the tabs will only match up correctly if all of the cards are flipped to their compatible sides.

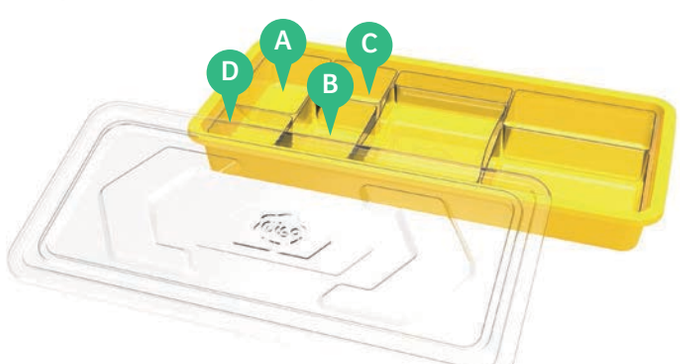
See page 11 for a **complete list of all the map cards** included in the kit and their functions.

Paper Card Storage Method Suggestion

We designed this helpful tray for you, so you can find and use your code cards easily. We suggest you divide the paper cards into these two categories:

Code cards: When being stored, they should be placed in the storage compartments marked A and B in the picture below. When needed they can easily be taken out and placed in the smaller boxes marked C and D.

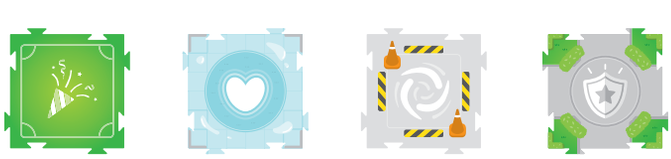
Map Cards: base map cards and code card frames can be kept in the large bucket area, to protect them from damage.



We have the following code cards and map cards in this package. If you are interested in learning more about conditional statements, event cards, and programming exercises, then buy the primary school manual.

Conditional Element (Conditional Statement Cards & Event Cards)

Event Map Cards



Basic Operation

Turning the Robot On and Off

1. Make sure that a parent or adult has installed the **batteries** as described on the inside front cover of this manual.
2. Slide the **power switch** located on the bottom of the robot base unit to the on position.
3. The robot will light up and play its **startup** sounds.
4. The robot is now **standing by for recording.** It will pulse with a blue light.
   When you are not using the robot, turn the power off by sliding the switch into the off position to save **battery energy** or take out the batteries for safety. Programs are erased when you turn the robot off.

If you don't use the robot for five minutes, it will automatically go into a **sleep mode**. Programs are preserved while the robot is sleeping. You can press either button to wake up the base unit.
When the batteries are running low, the robot will alert you with a flashing orange light and play a low-power indicator sound.

Recording a Program

You **program** the robot by **laying out a series of code cards** for the robot to drive over and record. Here's how it works:

1. Make sure the robot is **turned on** and standing by for recording.
2. **Lay out a series of code cards** in the code card frame(s). A main program can have up to 30 code cards, not including the Start and End code cards.

Subroutine programs, or **functions,** are introduced on page 4. Functions can have up to 15 code cards.
If your table is too short to place all of the frames in a row, no problem! **You can record any program in segments.** The robot won't stop recording until it scans the End code card. Therefore, you can scan one row of code, and the robot will pause at the end. Then you can move the robot to another row, and the robot will automatically continue recording.
If your robot scans the maximum number of cards but did not scan an End card, the robot will automatically end the program.

3. **Place the robot directly above the Start code card** at the start of the frame, facing toward the rest of the code.
4. **Press the Record button.**
5. The robot will pulse with red light, its Record button will pulse green, and it will play music indicating it is recording. At the same time, the robot will drive forward over the code cards, one at a time, **scanning and recording the program.**
6. For each successfully recorded code card, the robot will play a sound.
7. If the robot encounters any **problem** while recording, it will **flash orange and red** and play an **error sound.** This could happen if the robot is facing the wrong direction or if the code cards were laid out in an incorrect order.
8. When the robot reaches the End card and scans it, the robot will stop moving and play a **finished-recording** sound.
9. The robot will now be **standing by to run** its program. Its Record button light will now be solid green.
10. If there is a subroutine function to program, place the robot on the Function Start card and press the Record button. **The robot remembers one main program and up to three functions at one time in its memory.**

Running a Program

Once a program has been recorded, you can **run the program.**
Here's how:

1. **Place the robot on the Start map card**, facing the direction of the arrows.
2. **Press the Run button.** Record and Run are the same button. The robot knows whether to record or run based on whether it is sitting on a code card or a map card.
3. The robot will now **run the program.**
   The robot will first move around a little on the Start map card to **orient itself.** This is important so that it stays aligned to the map cards throughout the program.
   If at this time there is no main program recorded on the robot, it will flash between red and orange and play a warning sound.
   As the robot runs the program, it will play its running background music, unless the program tells it to play other music.
4. When the robot reaches and scans certain map cards, such as Event or Base map cards, it may trigger **special behaviors or functions.**

After running a program, the robot still remembers the programs; the **programs are not automatically erased** after running. You can **run the program again**, or record or overwrite the program or function.

Overwriting Programs

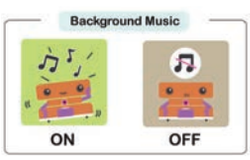
The robot can only hold one main program and one of each subroutine functions at a time. If you have the robot record a new program or function (starting with the Start card or one of the Function Start cards) when there is already a program or function saved, the robot will **overwrite the old main program or function.**
This means the old program is erased and the new one replaces it.
If you want to revise the main program or a function, you can overwrite them one at a time; the other programs are saved.

Erasing Programs

To **completely erase** all of the programs on the robot (and quit Math Mode), **press and hold the Erase button for two seconds or longer.**
The robot's light will flash red for a few seconds and then stop, indicating that the program memory has been cleared.

Background Music

If you want to **turn the background music off or on**, scan the Background Music code graphic on page 73. The background music is on by default. The graphic looks like this:



Lessons

The best way to learn what all the code cards do and how they work together is by following the **lessons** in this manual.

For each lesson, you **first build some models.**
The step-by-step assembly instructions are printed before the lessons in which they are used. Then, you **lay out the grid of map cards** exactly as shown in the lesson, and also the **series of code cards**. Then **record and run** the program and observe what the robot does. Did it all work perfectly? Congratulations! If not, you should go through a debugging process to fix the physical model, the code cards, and/or the map cards until it works perfectly!

Basic Operation

Code Card Definitions

Each code card itself represents a function or chunk of code that tells the robot's motors, light, and speaker what to do. Here are the specs for each code card and how many are included in the kit, counting both sides.

Image	Name Dedcriotion	Qty.
	**Start** Every main program must begin with this card. Only used in the main program.	4
	**End** Every program, including main and function programs, must end with this card.	10
	**Red Function Start** The red function program must start with this card. This function is called when the robot scans the matching base map card (red star).	2
	**Green Function Start** The green function program must start with this card. This function is called when the robot scans the matching base map card (green star).	2
	**Blue Function Start** The blue function program must start with this card. This function is called when the robot scans the matching base map card (blue star).	2
	**If (Conditional Element)** This is the start card for a conditional (if-then) function. When the robot scans an Event card that satisfies the condition, the function runs.	2
	**Do (Conditional Element)** This card can only be used with the If card in a conditional function. If the condition is satisfied, the sequence after the Do card runs.	2
	**Else (Conditional Element)** This card can only be used with the If card in a conditional function. If the condition is not satisfied, the sequence after the Else card runs. Note: Move Forward, Move Backward, Turn Right, Turn Left, and Pause Movement cannot be used in the Else statement after the Else card.	2
	**And (Conditional Element)** This card can only be used with the If card in a conditional function. When used, two conditions must be met for the function to run.	1
	**Or (Conditional Element)** This card can only be used with the If card in a conditional function. When used, either one of two conditions can be met for the function to run.	1
	**Event 1 (Conditional Element)** Used in a conditional function, this card defines the condition that must be met for the function to run. The robot must scan the matching map card.	1
	**Event 2 (Conditional Element)** Used in a conditional function, this card defines the condition that must be met for the function to run. The robot must scan the matching map card.	1
	**Event 3 (Conditional Element)** Used in a conditional function, this card defines the condition that must be met for the function to run. The robot must scan the matching map card.	1
	**Event 4 (Conditional Element)** Every Used in a conditional function, this card defines the condition that must be met for the function to run. The robot must scan the matching map card.	1

Image	Name Dedcriotion	Qty.
	**Green Simple Loop Start/End** These two cards allow you to repeat a sequence of code placed between them a specific number of times, defined by a number card.	4
	**Red Simple Loop Start/End** These two cards allow you to repeat a sequence of code placed between them a specific number of times, defined by a number card.	4
	**Move Forward** This card tells the robot to move forward one map card. It can only be used in the main program. It can be repeated with a number card	24
	**Move Backward** This card tells the robot to move backward one map card. It can only be used in the main program. It can be repeated with a number card.	24
	**Turn Right (Clockwise)** This card tells the robot to turn 90 degrees to the right. It can be repeated with a number card.	18
	**Turn Left (Counterclockwise)** This card tells the robot to turn 90 degrees to the left. It can be repeated with a number card.	18
	**Pause Movement** This card tells the robot to pause for one second. It can only be used in the main program. It can be repeated with a number card.	4
	**Turn Output Gear Clockwise** This card tells the robot to turn its output gear clockwise for one second. It can only be used in a function. It can be repeated with a number card.	5
	**Turn Output Gear Counterclockwise** This card tells the robot to turn its output gear counterclockwise for one second. It can only be used in a function. It can be repeated with a number card.	5
	**Pause Output Gear** This card tells the robot to pause turning its output gear for one second. It can only be used in a function. It can be repeated with a number card.	4
	**Play Sound: Hi!** This card tells the robot to make a "Hi!" sound. It can be repeated with a number card.	2
	**Play Sound: Ahh** This card tells the robot to make an "Ahh" sound, as if happy. It can be repeated with a number card.	2
	**Play Sound: Huh?** This card tells the robot to make a "Huh?" sound, as if questioning. It can be repeated with a number card.	2
	**Play Sound: Aargh** This card tells the robot to make an " Aargh" sound, as if frustrated. It can be repeated with a number card.	2
	**Play Sound: Mouse** This card tells the robot to squeak like a mouse singing a little song. It can be repeated with a number card.	2
	**Play Sound: Penguin** This card tells the robot to make the sound of a squawking penguin. It can be repeated with a number card.	2

If you're interested in learning more about how to use the Conditional Element Cards, please refer to the primary school manual for further information.

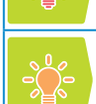
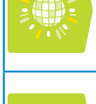
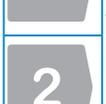
Image	**Name** Description	Qty.
	**Play Sound: Cheering** This card tells the robot to play the sound of a cheering crowd. It can be repeated with a number card.	2
	**Play Sound: Factory** This card tells the robot to play the sound of machines in a factory. It can be repeated with a number card.	2
	**Play Sound: Fire hose** This card tells the robot to play the sound of a fire hose spraying water. It can be repeated with a number card.	2
	**Play Sound: Siren** This card tells the robot to play the sound of an emergency vehicle's siren. It can be repeated with a number card.	2
	**Light Color: Blue** This card tells the robot to change the color of the light inside its output gear to blue for one second. It can be repeated with a number card.	2
	**Light Color: Purple** This card tells the robot to change the color of the light inside its output gear to purple for one second. It can be repeated with a number card.	2
	**Light Color: Pink** This card tells the robot to change the color of the light inside its output gear to pink for one second. It can be repeated with a number card.	2
	**Light Color: Red** This card tells the robot to change the color of the light inside its output gear to red for one second. It can be repeated with a number card.	2
	**Light Color: Orange** This card tells the robot to change the color of the light inside its output gear to orange for one second. It can be repeated with a number card.	2
	**Light Color: Yellow** This card tells the robot to change the color of the light inside its output gear to yellow for one second. It can be repeated with a number card.	2
	**Light Color: Green** This card tells the robot to change the color of the light inside its output gear to green for one second. It can be repeated with a number card.	2
	**Light Color: Rainbow** This card tells the robot to cycle through seven colors of light in its output gear for half a second each. It can be repeated with a number card.	2
	**Light Effect: Disco Strobe** This card tells the robot to light up its output gear in a very-fast, on-off flashing pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Emergency Vehicle Light** This card tells the robot to light up its output gear in a pattern like an emergency vehicle's light, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Falling Star** This card tells the robot to light up its output gear in a fast-slow-fast flashing pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Twinkling Star** This card tells the robot to light up its output gear continuously with a little twinkle in the middle, in a purple color by default, and for three seconds. It can be repeated with a number card.	2

Image	**Name** Description	Qty.
	**Light Effect: Firefly** This card tells the robot to light up its output gear in a pattern like a firefly's light, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Slow Blink** This card tells the robot to light up its output gear in a slow, on-off blinking pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Medium Blink** This card tells the robot to light up its output gear in a medium-speed blinking pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Fast Blink** This card tells the robot to light up its output gear in a fast, on-off blinking pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Speeding Up** This card tells the robot to light up its output gear in an increasingly fast blinking pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Light Effect: Slowing Down** This card tells the robot to light up its output gear in a decreasingly fast blinking pattern, in a purple color by default, and for three seconds. It can be repeated with a number card.	2
	**Number Cards 1 through 9** These cards tell the robot to repeat a code card's instructions by the number of times printed on the number card when the number card is placed immediately after the code card in the sequence. This only works with the following code cards: • Simple Loop Start (but not Simple Loop End) • Move Forward and Move Backward • Turn Right and Turn Left • Pause Movement • Turn Output Gear Clockwise and Turn Output Gear Counterclockwise • Pause Output Gear • All Play Sound cards • All Light Color cards • All Light Effect cards You cannot place more than one number card consecutively (one after another without interruption) in a program, or the robot will give you an error. • Number 1: Execute preceding code card 1 time • Number 2: Execute preceding code card 2 times • Number 3: Execute preceding code card 3 times • Number 4: Execute preceding code card 4 times • Number 5: Execute preceding code card 5 times • Number 6: Execute preceding code card 6 times • Number 7: Execute preceding code card 7 times • Number 8: Execute preceding code card 8 times • Number 9: Execute preceding code card 9 times	2
		2
		2
		2
		2
		2
		2
		2
		2

Basic Operation

Map Card Overview

There are four basic types of map cards included in this kit. The map cards are not all interchangeable, as each one has a special invisible pattern printed on it. You need to make sure you are using the correct map cards in the correct places. Here is an overview of all the map cards.

Start Map Cards

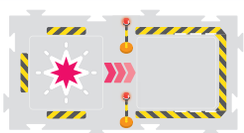
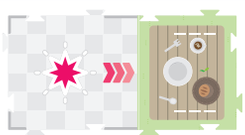
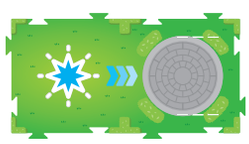
Front	Back
	

Event Map Cards*

Front	Back
 Event 2	 Event 4
 Event 3	 Event 1

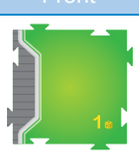
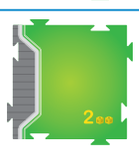
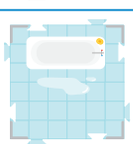
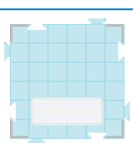
If you're interested in learning more about how to use the Conditional Element Cards, please refer to the primary school manual for further information.

Base Map Cards

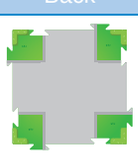
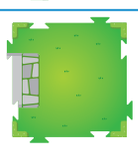
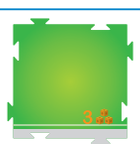
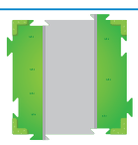
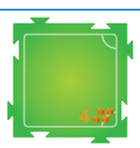
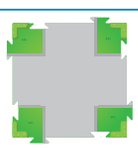
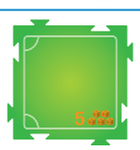
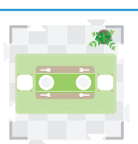
Front	Back
 Red Function Base Map Card Front	 Red Function Base Map Card Back
 Green Function Base Map Card Front	 Green Function Base Map Card Back
 Blue Function Base Map Card Front	 Blue Function Base Map Card Back

*Note: Event map cards cause the robot to perform a default action when they are scanned and no matching event code card has been used.

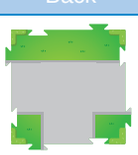
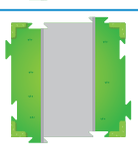
General Map Cards

Front	Back
	
	
	
	
	

General Map Cards

Front	Back
	
	
	
	
	

General Map Cards

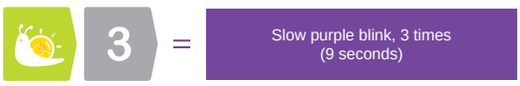
Front	Back
	
	
	

Combining Light Cards

The light color and light effect code cards can be combined together in the program to make more complex results.

Place a number card after a light color card to change the number of times the light color runs, making the light stay on longer.

Place a number card after a light effect card to change the number of times the light effect runs, making the light effect run longer.

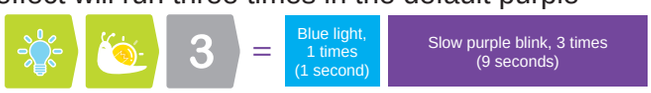


Examples

If you combine the cards together as follows, the light effect runs two times and then the light changes to blue and runs three times, which is about three seconds in this case.

If you combine the cards together as follows, the color of the light effect will be blue instead of the default purple and it will run three times, or about nine seconds in this case. When the light effect card comes before the light color card, the color of the light effect changes.

If you place the cards as follows, the light effect will not be combined with the light color. The blue light will shine for one second and then the light effect will run three times in the default purple color. When the light effect card comes after the light color card, the two cards do not combine.



Not sure how to interpret all this? When in doubt, try it out!

Troubleshooting Tips

If your robot isn't recording:

• Make sure you are starting your program with a Start, Function Start, or If code card.
• Make sure your robot's batteries are charged and the robot is not giving you the low-power indicator alert.
• Make sure the robot is facing the correct direction, following the arrows on the code card frame.

If your robot is acting funny or not working properly:

• Make sure the batteries are sufficiently charged. When the batteries are running low, the robot will alert you with a flashing orange light and play a low-power indicator sound.
• Dust, stains, or fading on the surface of the invisible pattern cards may interfere with the reading of the invisible pattern codes. Please keep the cards clean and dry.
• If your robot can't record a function start code card or an If code card, the robot might be in math mode. Hold down the Erase button for two seconds to go back to normal mode.

If your robot is flashing orange and stopping
in the middle of a line of code cards:

• If the robot encounters any problem while recording, it will flash orange and red and play an error sound. Check the code cards and make sure they are in the correct order.

If your robot does unexpected movements when it starts to run a program:

• This is normal. The robot is calibrating its position. If you place the robot in the center of the Start map card, it will have a shorter calibration time.
• The calibration process improves the precision of the robot's movements. Do not move the robot during the calibration process.
• The robot runs a quick calibration when you press the Run button. The robot runs a standard calibration the first time you press the Run button after turning the robot on, or when you press and hold the Run button for two seconds.
The main program is executed after calibration.

Parts List


1 x5
2 x30
3 x5
4 x1
5 x4
6 x2
7 x2
8 x1
9 x1
10 x2
11 x2
12 x1
13 x2
14 x2
15 x16
16 x20
17 x8
18 x8
19 x8
20 x12
21 x4
22 x2
23 x18
24 x12
25 x6
26 x12
27 x4
28 x16
29 x4
30 x2
31 x4
32 x4
33 x4
34 x4
35 x6
36 x12
37 x2
38 x1
39 x1
40 x8
41 x1
42 x1
43 x1
44 x1
45 x1
46 x1
47 x1
48 x1
49 x1
50 x1
51 x1
52 x1
53 x1
54 x1
55 x1
56 x1

Parts List:

No.	Description	Item No.	Qty.
1	C-AXLE	7026-W10-H1K	5
2	B-SHORT PEG	7344-W10-C2B	30
3	C-20mm AXLE CONNECTOR	7413-W10-T1B	5
4	C-WORM GEAR	7344-W10-A1S1	1
5	C-20T GEAR	7026-W10-D2K	4
6	C-60T GEAR	7026-W10-W5B	2
7	C-MOTOR AXLE	7026-W10-L1D	2
8	C-60mm AXLE II	7413-W10-M1D	1
9	C-100mm AXLE II	7413-W10-L2D	1
10	C-3 HOLE ROUND ROD	7404-W10-C1W	2
11	C-7 HOLE ROUND ROD	7404-W10-C2W	2
12	C-7 HOLE PROLATE ROD	7404-W10-C3W	1
13	C-5x5 ARCH FRAME	7411-W10-F1K	2
14	C-5x5 ARCH FRAME	7411-W10-F1W	2
15	B-CUBE	880-W10-A1W	16
16	B-CUBE	880-W10-A1B2	20
17	B-CUBE	880-W10-A1K1	8
18	B-CUBE	880-W10-A1G1	8
19	B-CUBE	880-W10-A1P1	8
20	B-CUBE	880-W10-A1O3	12
21	B-6 HOLE CUBE	880-W10-N1W	4
22	B-6 HOLE CUBE	880-W10-N1O3	2
23	B-CONVEX	880-W10-R1W	18
24	B-CONVEX	880-W10-R1B1	12
25	B-CONVEX	880-W10-R1K1	6
26	B-CONVEX	880-W10-R1YG	12
27	B-CONVEX	880-W10-R1P	4
28	B-CONVEX	880-W10-R1O3	16
29	B-TRIANGLE	880-W10-S1W	4
30	B-TRIANGLE	880-W10-S1B2	2
31	B-TRIANGLE	880-W10-S1K1	4
32	B-TRIANGLE	880-W10-S1O3	4
33	B-CONCAVE	880-W10-D1W	4
34	B-CONCAVE	880-W10-D1B2	4
35	B-CONCAVE	880-W10-D1O3	6
36	C-SHORT BUTTON FIXER	7061-W10-W1W	12
37	C-OD8x30mm TUBE	7400-W10-G1D	2
38	B-GLOBAL PIECE	7128-W10-E1K	1
39	B-4-SIDED PYRAMID PIECE	7128-W10-E4O1	1
40	B-EYE	7128-W22-2	8
41	B-PEG REMOVER	7061-W10-B1Y	1
42	C-30mm AXLE CONNECTOR	7413-W10-U1S	1
43	C-SAMMY'S CRUST, BOTTOM	7442-W10-G2T1	1
44	C-SAMMY'S CRUST, TOP	7442-W10-G1T1	1
45	C-SAMMY'S GEARBOX, TOP	7442-W10-F1T1	1
46	C-SAMMY'S GEARBOX, BOTTOM	7442-W10-F2T1	1
47	C-SAMMY'S ARM, LEFT	7442-W10-H1P	1
48	C-SAMMY'S ARM, RIGHT	7442-W10-H2P	1
49	P-SAMMY'S EYE STICKER SHEET	R20#7442	1
50	P-DIE-CUT GRAPHICS SHEET	K16#7442	1
51	P-MAP CARD STRAPS	K41#7442	1
52	P-MAP CARDS	K16#1276-1	1
53	P-BASE MAP CARDS	K16#1276-2	1
54	P-CODE CARDS	K16#1276-3	1
55	P-CODE CARD FRAMES	K16#1276-4	1
56	C-ROBOTIC BASE UNIT	7442-W85-A	1

Tips and Tricks:

Here are a few tips for assembling and using the models.
Read them carefully before starting.

For more assembly tips, please refer to

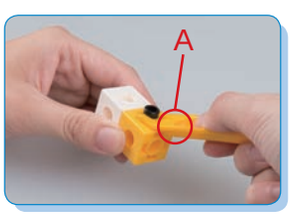


Figure 1. SHORT PEGS

Use the peg remover to pry short pegs out, as shown in Figure 1.

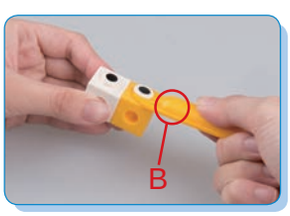


Figure 2. EYES

Use the peg remover to pry the eyes out, as shown in Figure 2.

Peanut Butter and Jelly Coding

Why is Sammy shaped like a peanut butter and jelly sandwich? Sammy was inspired by a classic activity that is used to introduce students to computer science. In this activity, students are asked to write a program, or a series of instructions, for another student or the instructor to follow to make a peanut butter and jelly sandwich.

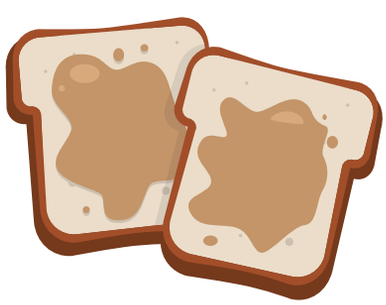
This activity teaches many concepts from **computer science**. Students learn to write precise and thorough instructions. They learn how computers do only what they are programmed to do. And they learn about the process of debugging, or the repetitive process of finding errors in a program, correcting them, and then retesting the program.

You can do a simple version of this activity here.

Obviously, don't attempt this if you or your child have any allergies or dietary restrictions that would cause any issues with it.

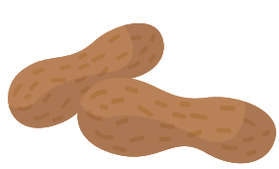
You will need:

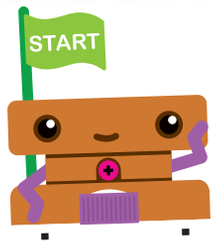
Package of sliced bread, jar of peanut butter, jar of jelly, dull knife, plate, paper, pen or pencil



Here's how:

1. Set out and review the materials needed to make a peanut butter and jelly sandwich with your child.
2. Ask your child to tell you the steps to making a peanut butter and jelly sandwich. Write down each step as your child dictates them to you.





3. When you are done writing the instructions, start following the instructions, one step at a time.
4. Take the instructions as literally as possible. For example, if the instruction is to put the peanut butter on the bread, you might take the jar of peanut butter and put it on the package of bread. If the instruction is to pick up the knife, you could pick it up by the blade instead of the handle. If an instruction is impossible to perform, you could freeze up and state that an error has occurred.
5. Go through the instructions step by step and try to debug them with your child. In the end, you should have a peanut butter and jelly sandwich that you can eat together as a snack.

This activity is a simple model for demonstrating how a computer program works to control a robot.

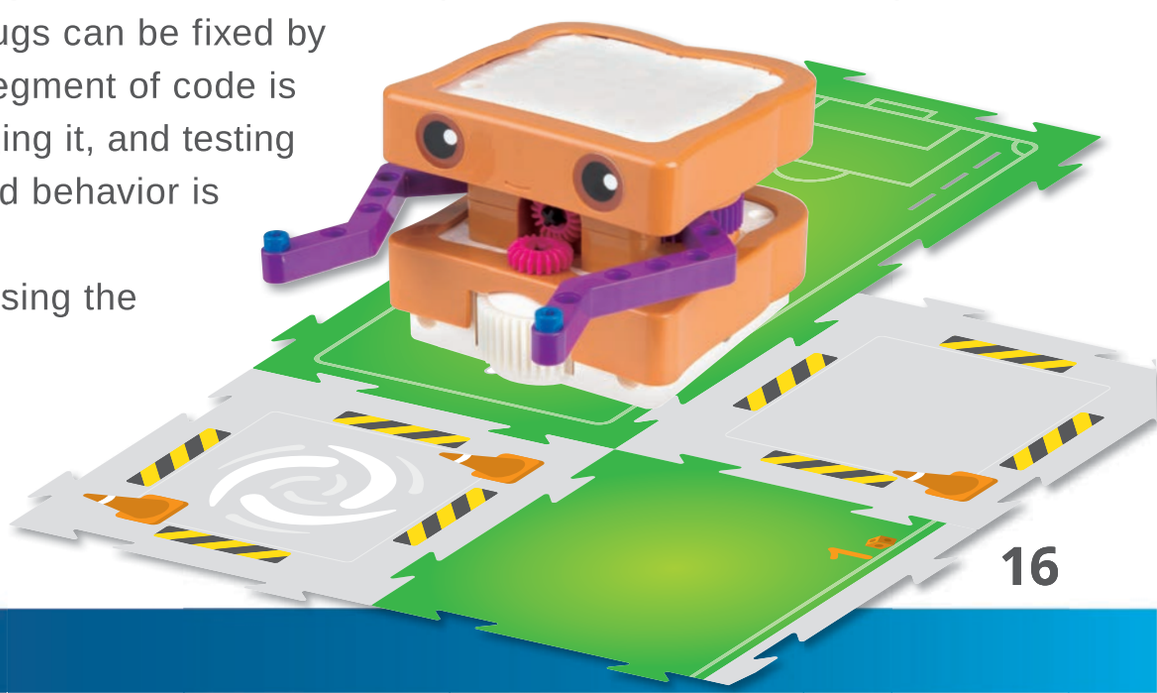
The list of instructions is the computer program and you played the role of the robot.

As you probably observed, a program's instructions must be performed in a **sequence**, or a specific order.

They must be written in a language, or **code**, that the robot's computer can understand. They must describe everything that the programmer wants the robot to do. The robot will not do anything other than what is programmed.

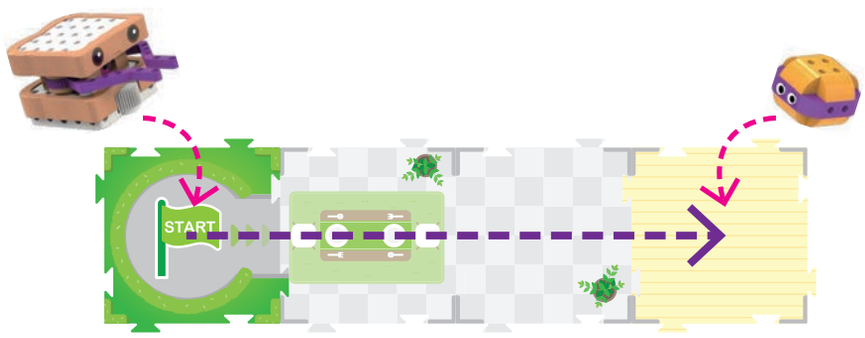
In the process of programming and testing programs, it is usually the case that the program behaves in an unexpected or unintentional way. This is a **bug.** Bugs can be fixed by locating which segment of code is causing it, changing it, and testing it until the desired behavior is achieved.

Now, let's start using the robot!



1 Sammy Visits Hammy

Learning Goal: Robot



Sammy is a robot. Robots come in all shapes and sizes. Sammy happens to be shaped like a peanut butter and jelly sandwich! Sammy has wheels powered by an electric motor inside its robotic base unit. Sammy also has purple arms connected to gears that are connected to an electric motor.

From Lesson 1 to Lesson 5, you will build and program Sammy to visit other food-friends that live in Foodville. First, Sammy is going to visit Hammy in Hammy's house. Sammy has to pass through two rooms to get to the room Hammy is in. Can you program Sammy to do this?

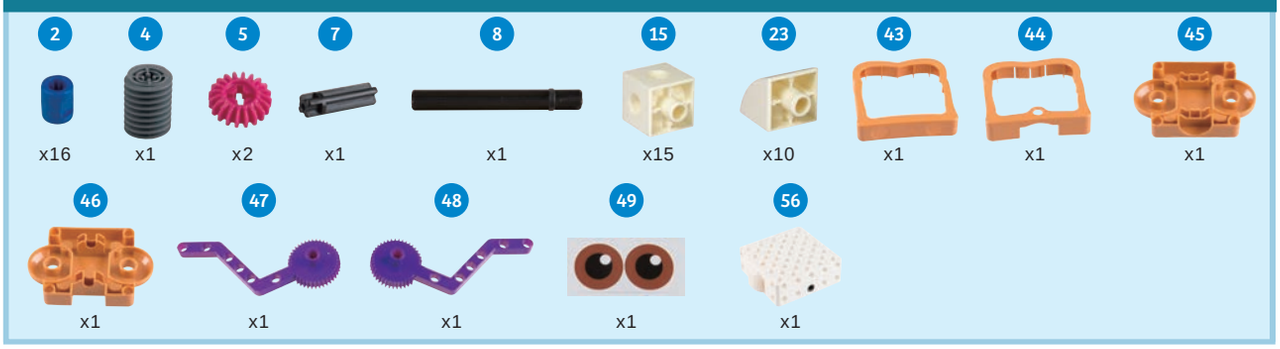
CHECK IT OUT

A robot is a machine controlled by a computer program that is programmed to perform various tasks and actions. These tasks include assembling cars, playing soccer, cleaning the floor, delivering parcels, drawing maps, climbing mountains, entertaining people, cooking and countless other actions. The robot can sense the environment with sensors, and interacts with the environment with motors, lights, speakers and/or other output devices.

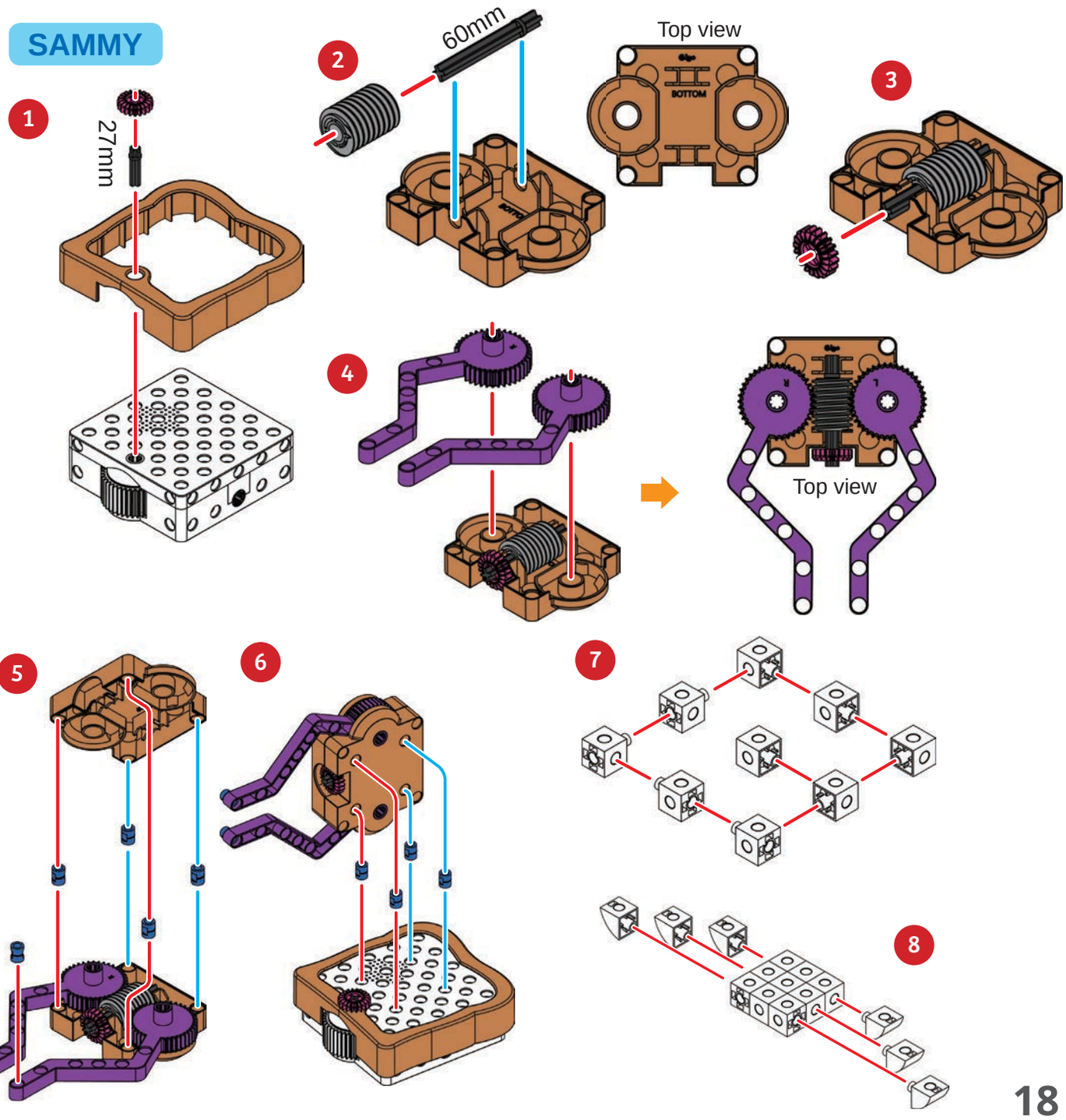
Brainstorming

Robots are used almost everywhere today. Talk about one type of robot you know and explain what it does.

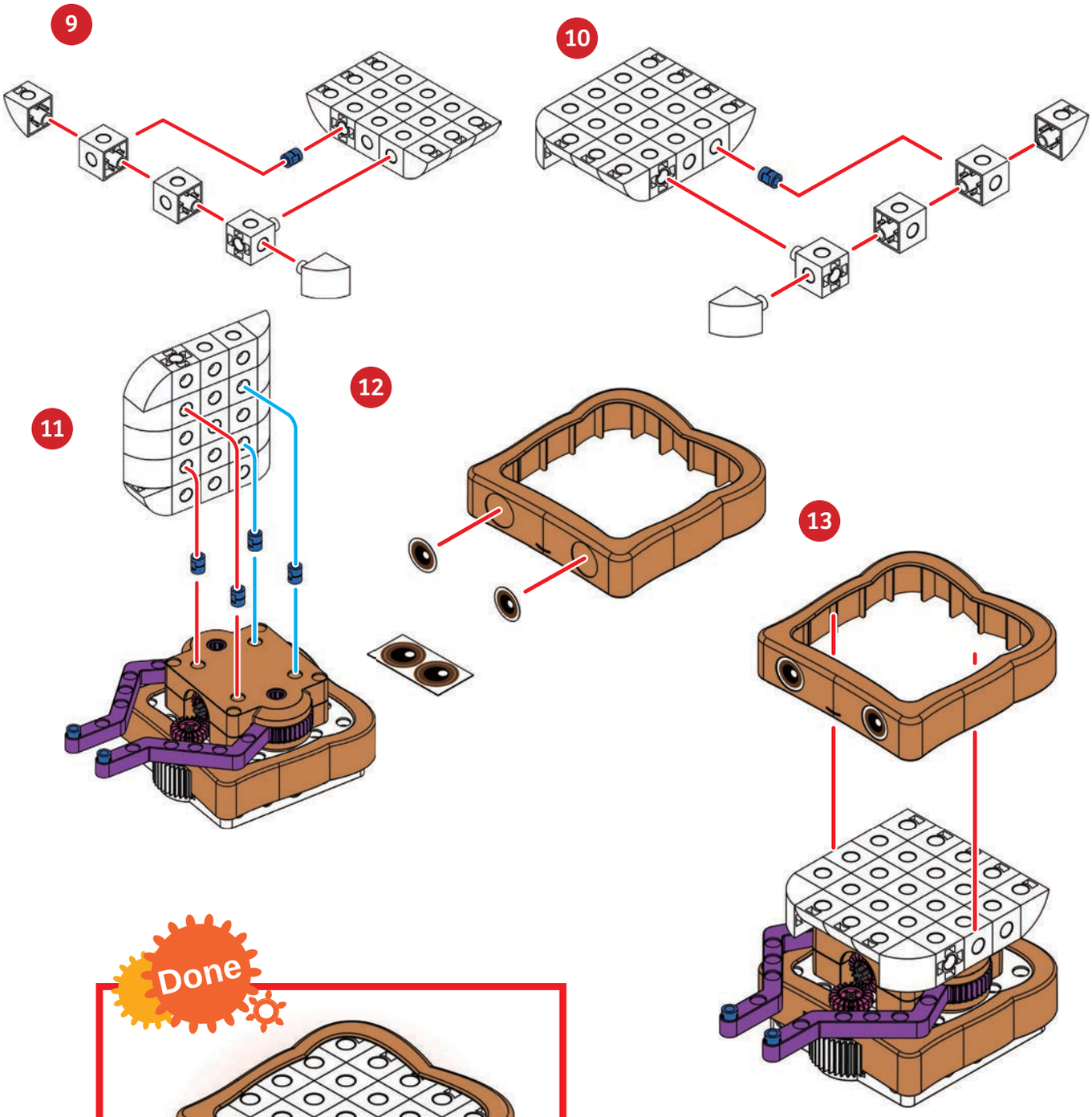
Parts List



SAMMY

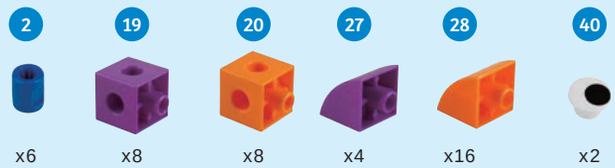


1 Sammy Visits Hammy

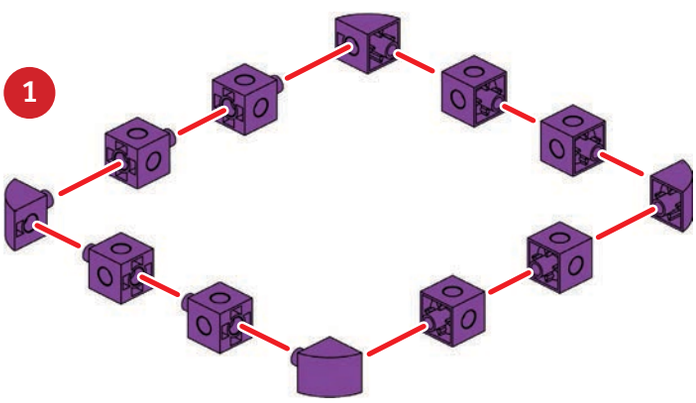


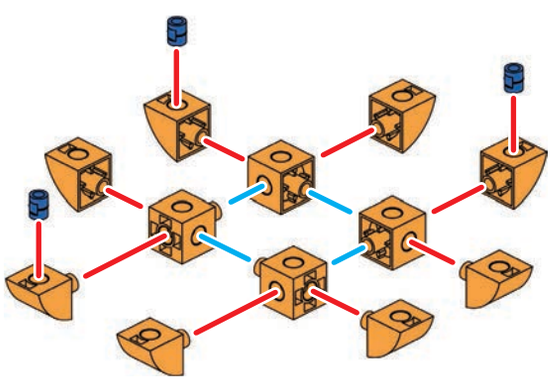
Smart Manual
Web Service

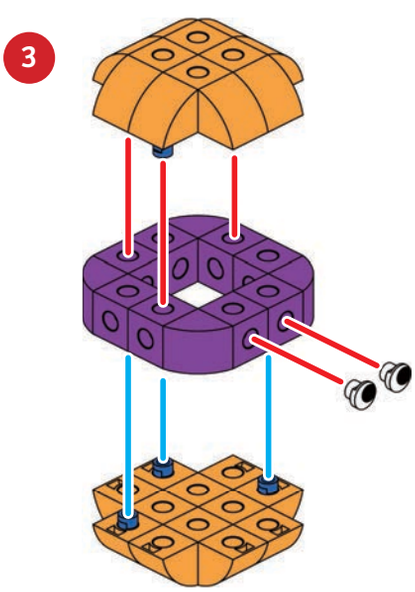
Parts List

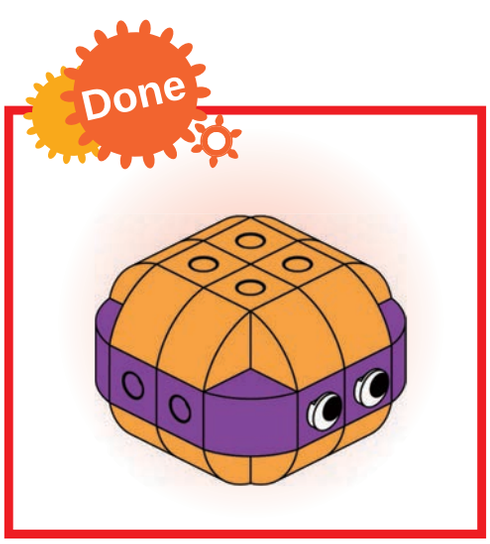


HAMMY









Smart Manual
Web Service

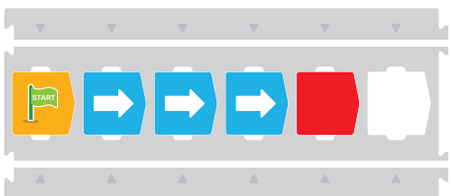
1 Sammy Visits Hammy

HERE'S HOW

Before starting, make sure you have read the introductory instructions for using the robotic base unit on pages 3 through 8.

1. Set up the map cards as shown. Place Hammy on the map card as shown.
2. Put the code cards into the code card frame in the order shown.
3. Turn Sammy on with the switch on the bottom.
4. Place Sammy on the Start code card. (You can align the axle hole of the robot's wheels with the dark gray arrows on the code card frame.) Press the Record button. Wait for Sammy to finish recording the program.
5. Place Sammy on the Start map card. Press the Run button (which is the same as the Record button).
6. Watch Sammy drive through the house and get to Hammy. Did everything work as you expected?

[CODE]



Put the **code cards** into the **code card frame** in this order.

What's Happening?

The robot scans a Start code, then three Move Forward code cards, then an End code card.
This results in a simple program that moves Sammy forward three map cards. Note how the robot always moves around a little to orient itself on the Start map card before running the program.

Hands-on Creativity

Try it: Add a map card to the map on this lesson and place Hammy on the last map card. Write a program to make Sammy visit Hammy successfully.

Evaluation

Model Operation Video

1 Model Assembled

2 Experiment Complete

3 Model Creation

2 Franky's Wake-Up Call

Learning Goal: programming language

Now Sammy is going to wake up Franky, who has overslept. Being a hot dog, naturally Franky's house is longer than Hammy's.
Program Sammy to drive into the house to get to Franky and then drive back outside again.

CHECK IT OUT

A programming language is a formal language, which comprises a set of instructions used to produce various kinds of output. Programming languages are used in computer programming to create programs that implement specific algorithms.
The earliest programming language was created before the invention of the computer. It was used to control the Jacquard Loom and Player Pianos.
When people communicate by language, the content can be unclear or even include some small mistakes. Usually the person listening can still understand what the person is trying to say. However, the computer listens differently. What a computer does is exactly "what it is told to do" and it cannot understand why the programmer writes the code in this way.

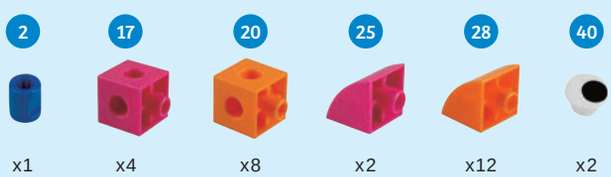
Brainstorming

First, try to talk about this lesson's path in your own language and write it down. Then, make a comparison between the path you write and the program in the manual.

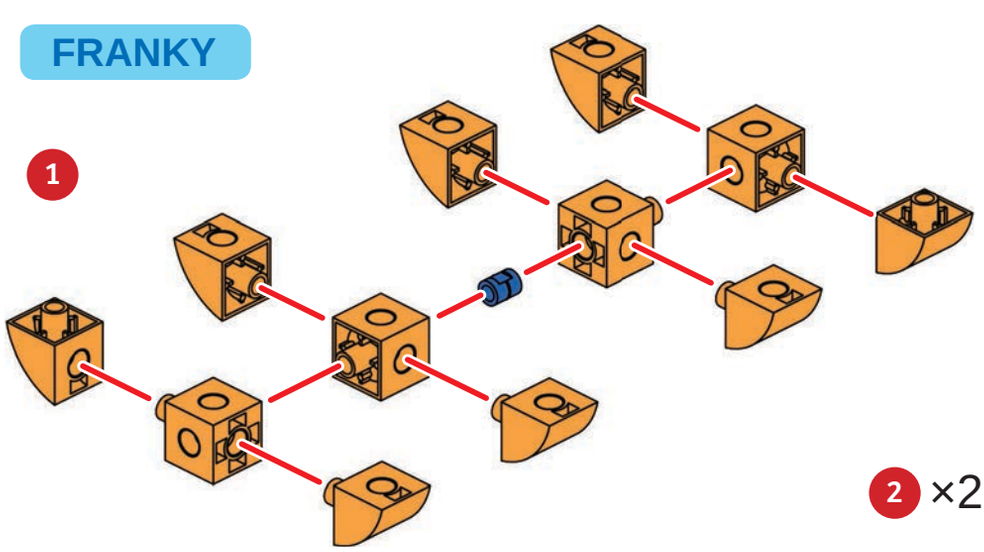
Sammy 's part list & assembly steps:

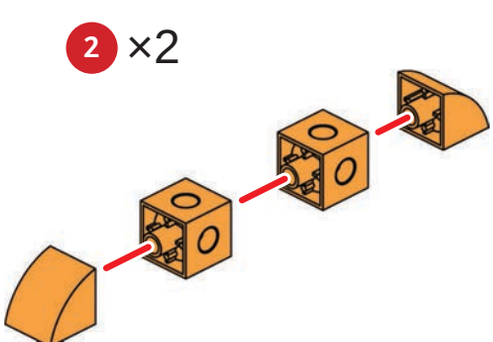
Please refer to Lesson 1.

Parts List



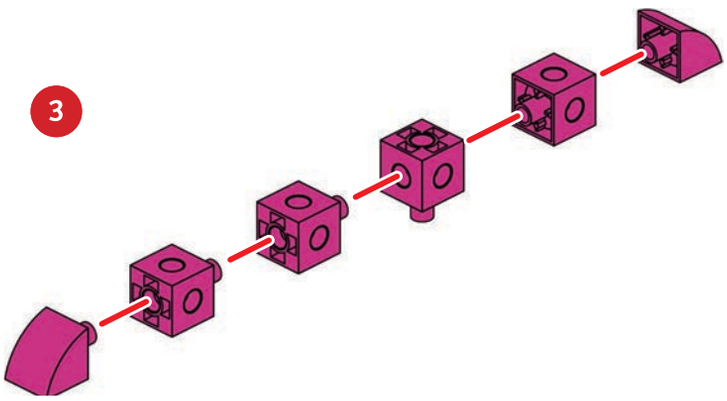
FRANKY



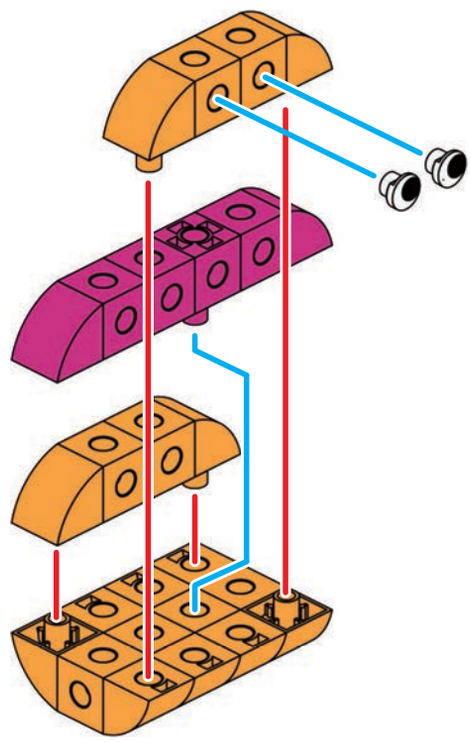


2 Franky's Wake-Up Call

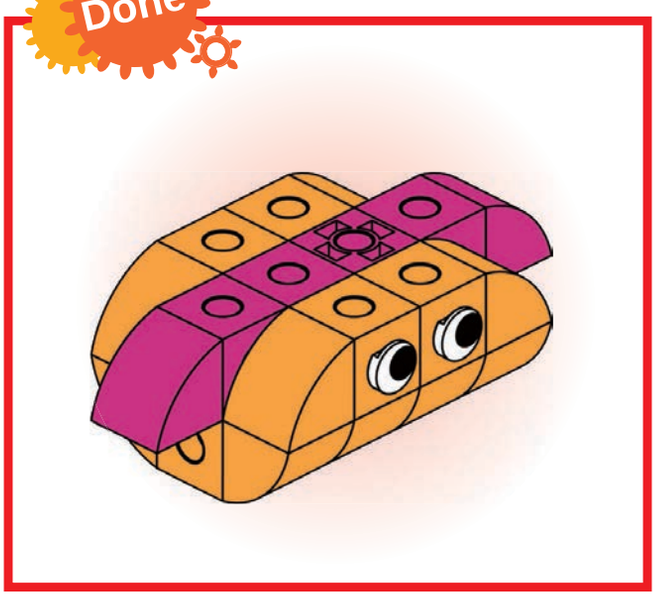
3



4

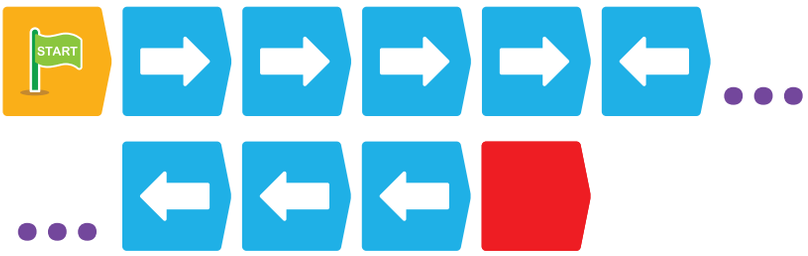


Done



Smart Manual
Web Service

[CODE]



Note: The dots here mean the program is continued on the next line because it was too long to fit on one line.

What's Happening?

This program uses a sequence of four Move Forward cards and four Move Backward cards. This results in a program that moves Sammy forward four map cards and then backward four map cards.

Hands-on Creativity

Try it: Write a program to let Sammy go to Franky's room, wake up Franky, and then return to the start following the same path.

Evaluation

Model Operation Video

1 Model Assembled

2 Experiment Complete

3 Model Creation

3 Turning a Corner

Learning Goal: Directionality

Waking up Franky has made Sammy exhausted! Sammy wants to go home to bed, which is around a corner in its house. What's the shortest program you can write to get Sammy there?

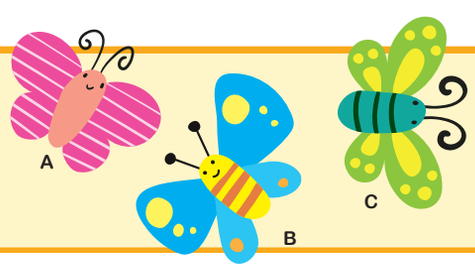
CHECK IT OUT

An object occupies a certain position in space. There is a mutual spatial relationship with an object to its surrounding. This is called the "spatial orientation" of the object.

Robots also have spatial orientation like humans. A robot travels on the map and interacts with other characters. The robot must be controlled by commands so that the robot can move in the correct direction and complete the programmed task. Therefore, it is necessary for the children to put themselves into the role of the robot in order to judge where to go based on the egocentric representations of the robot.

Brainstorming

Which is the right hand side of each butterfly, from the point of view of the butterfly?



Sammy 's part list & assembly steps:

Please refer to Lesson 1.

[CODE]

What's Happening?

In this program, you are using the Turn Left code card for the first time. First, Sammy moves forward three map cards. Then, the Turn Left code card rotates Sammy 90 degrees (a quarter of a full circle) so it is facing the bedroom. Finally, the last Move Forward card moves Sammy into the bedroom.

Hands-on Creativity

Try it: If you change the position of the room to the bottom of the path, how would you rewrite the program to make Sammy go to her room?

Evaluation

Model Operation Video

1 Model Assembled

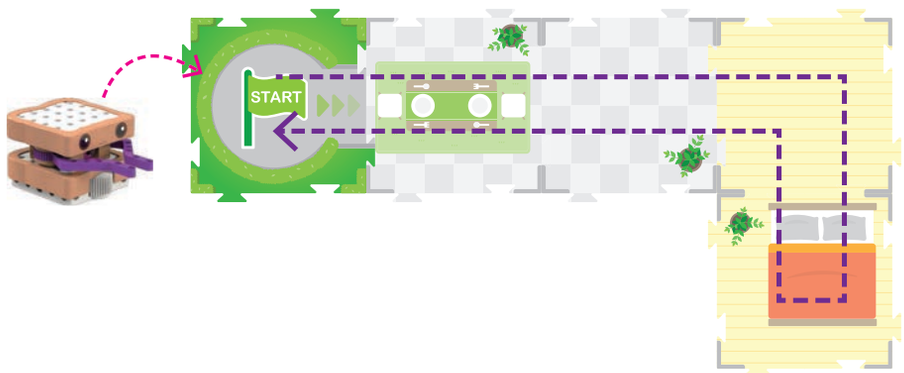
2 Experiment Complete

3 Model Creation

4 Touring a New Home

Learning Goal: Function

Sammy wants to take a tour of another house, which has a different layout than Sammy's own home. Can you write a program to move Sammy through the entire house and then back to the Start map card again?



CHECK IT OUT

A set of steps or commands arranged in a specific order. Computers run through the steps of a sequence in order, executing one at a time, for the purpose of performing a specific task that the sequence was created to perform.

Some tasks in life have to be done step by step. For example, if you want to wash hands, you have to turn on the faucet, wet your hands, turn off the faucet, soap your hands, rub the hands, turn on the faucet, rinse off the soap, rinse the soapy foam off the faucet, turn off the faucet and dry your hands.

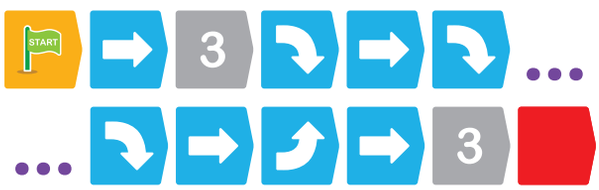
Brainstorming

In addition to washing your hands, please give other examples of things that need to be done in a sequence.

Sammy 's part list & assembly steps:

Please refer to Lesson 1.

[CODE]

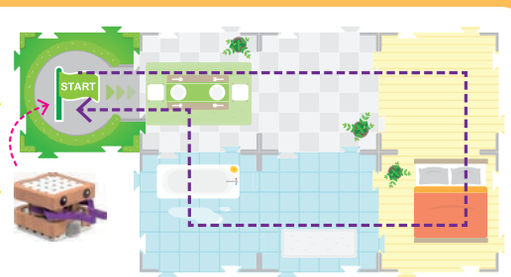


What's Happening?

You will use the Number Cards in this lesson. The number cards execute the code card immediately before them by the specified number of times. The first Number 3 card executes the Move Forward action three times. Sammy turns right and moves forward one square into the bedroom of the house. Then, she turns left twice (90x2=180 degree), she leaves the bedroom and turns left again. The second Number 3 card executes the Move Forward action three times to let Sammy goes back to the start. The code for this lesson is shorter because of the Number Cards.

Hands-on Creativity

Try it: Add two map cards to the map for this lesson as shown in the picture below. From the start position, program Sammy to move to the bedroom and back to the start following the purple path. How would you write this new program?



Evaluation

Model Operation Video

1 Model Assembled

2 Experiment Complete

3 Model Creation

5 Monograph 1

Sammy wants to visit two friends that are currently located at different spots around town. Place 1 or 2 Gigi Fish and Hammy on the map as shown.

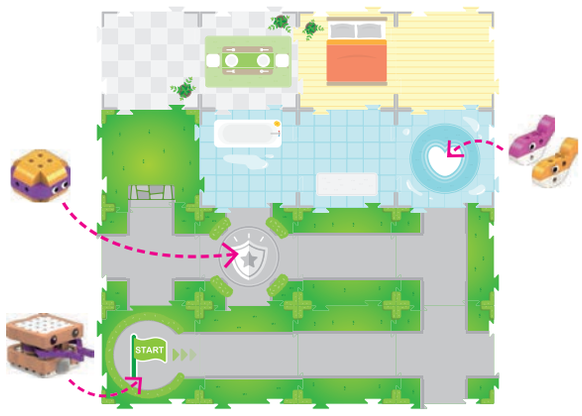
Can you write a program to make Sammy move around the map to visit both of them?

Model Review

1. Sammy

2. Hammy

3. Gigi (Fish)



4. Map

Design Concept

My Artwork

Evaluation

1 Model Design

2 Model Creation

3 Winner!

6 Pippy is Loopy for Cheese

Learning Goal: Loop

Pippy is a mouse who loves cheese. She is always trying to find cheese that people have left out.

Pippy is looking for a yummy piece of cheese. She thinks there might be one on the table in the dining room. Place the cheese on the dining room map card. Can you code Pippy to find the cheese?

What is the fewest number of code cards you can use to get Pippy to the cheese? Try using only Move Forward cards, number cards, and/ or simple loop cards.

CHECK IT OUT

The loop is a term used in computer science. A loop is a set of steps that commands a computer, robot or machine to repeat the steps a number of times. For example, a teacher says, "clap this rhythm 3 times", students will clap the rhythm 3 times and then stop. The phrase "clap the rhythm 3 times" is a type of loop. This loop is also called a count loop, because the loop is given a specific number of times.

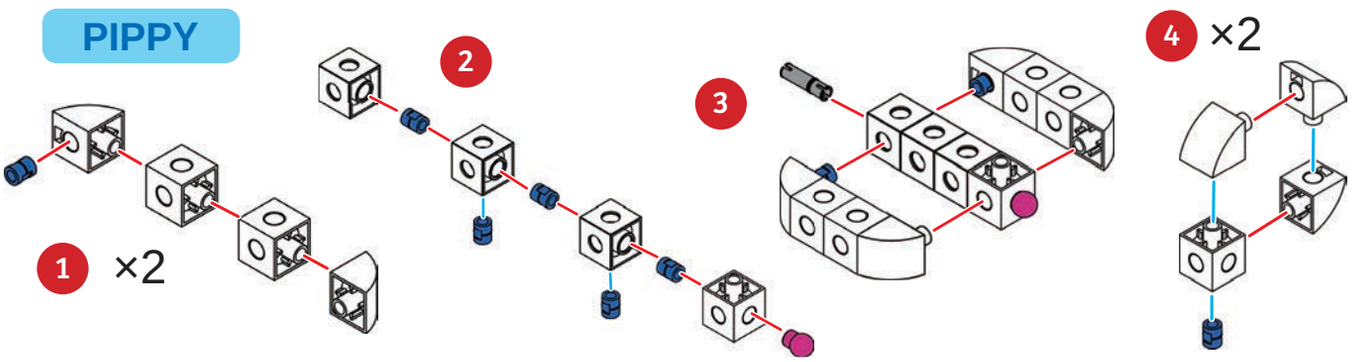
The loop can also be set to repeat forever (infinite loop or endless loop), or set to execute the program only when a certain condition occurs (while loop), or set to execute the program repeatedly until another condition occurs (conditional loop).

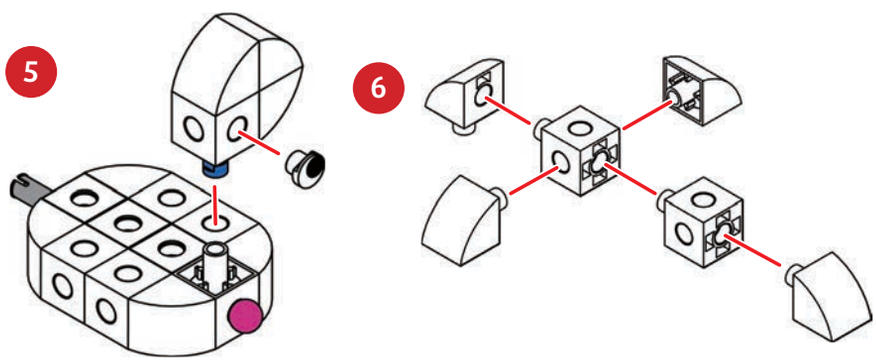
Brainstorming

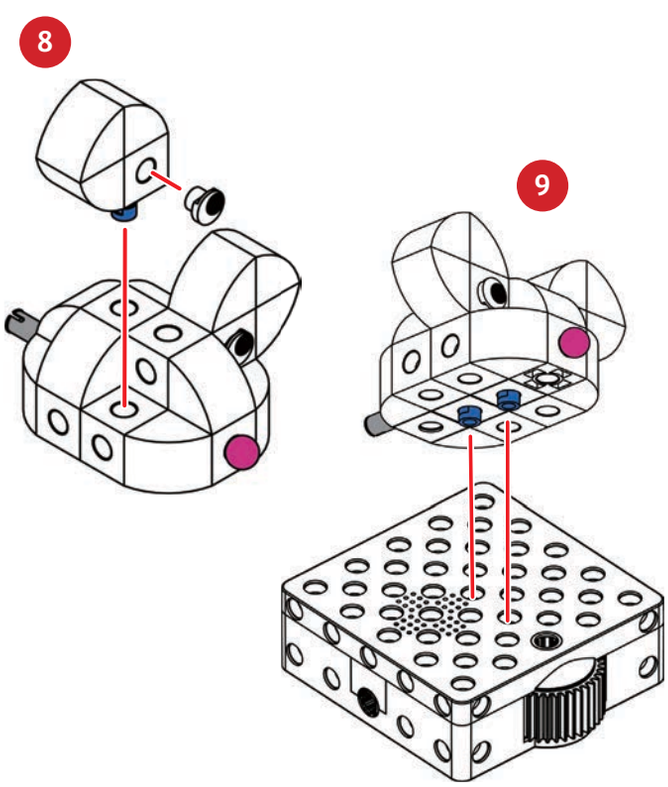
If a teacher says, “clap this rhythm until I raise my hands above my head and make a circle." How many times will the students repeat the rhythm?

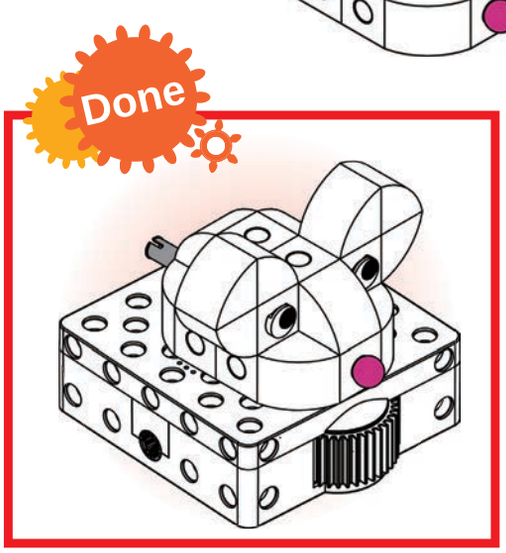
Parts List

PIPPY





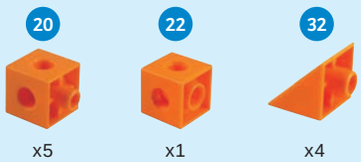




Smart Manual
Web Service

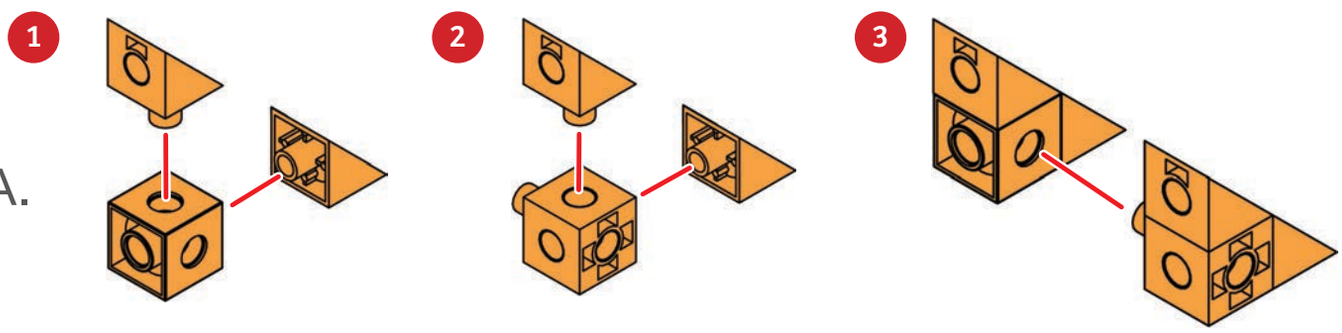
6 Pippy is Loopy for Cheese

Parts List

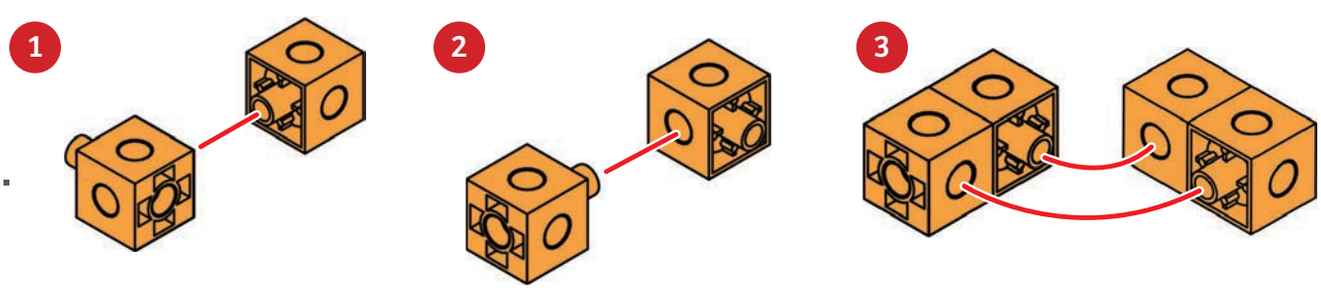


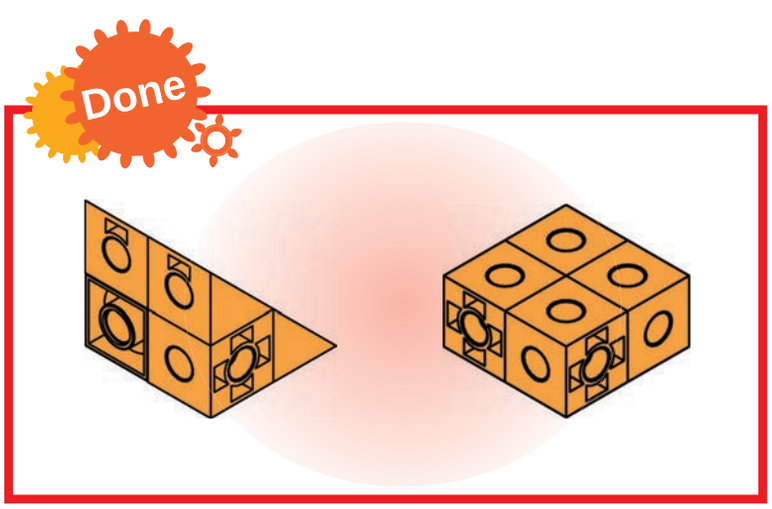
TWO CHEESES

A.



B.





A

Smart Manual
Web Service

B

Smart Manual
Web Service

[CODE]

A. 

B. 

C. 

D. 

What's Happening?

A. In this example, five Move Forward cards make Pippy move forward five map cards to the cheese.
B. In this example, the Number 5 card executes the Move Forward card five times, bringing Pippy to the cheese.
C. In this example, the Green Simple Loop is executed five times because of the Number 5 card. The loop is defined as one Move Forward card, so Pippy is moved forward five map cards to the cheese.
D. In this example, the Green Simple Loop is only executed once, but the Number 5 card repeats the move forward command five times.

Hands-on Creativity

Try it. Remove two map cards, and place the cheese at the end of the path. Write this program with loop cards.

Evaluation

Model Operation Video

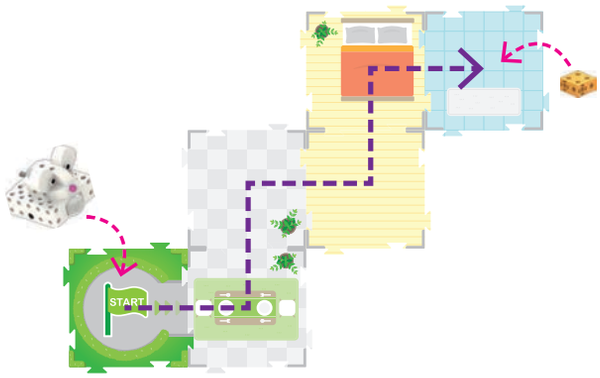
1 Model Assembled

2 Experiment Complete

3 Model Creation

7 Zig-Zag to the Cheese

Learning Goal: Debugging



Again, Pippy is looking for cheese. For some reason, the cheese is in the bathroom this time. Can you write a code to bring Pippy to the cheese? Can you use a loop to do it efficiently?

CHECK IT OUT

Debugging means to find the error. When a program, computer, or robot behaves in a way that does not correspond with the programmer's purpose, the programmer will start debugging, which means to find and resolve bugs or defects.

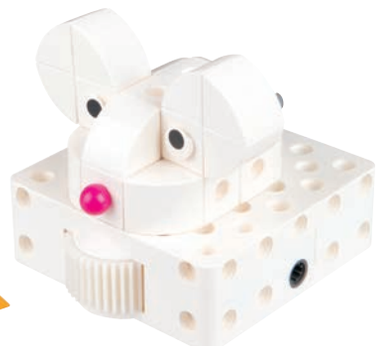
Debugging is part of programming.
For example, look at this sentence: The sun rises from the west. Which part of this sentence is incorrect? The process of finding the error is called debugging!

Brainstorming

Please find the error in the following examples and correct it.
Example: Gogo's younger brother really wants to go to the zoo. He puts on his shoes, then puts on his socks and leaves happily.

Pippy & two cheeses 's part list & assembly steps:

Please refer to Lesson 6.



[CODE]

A. 

B. 

What's Happening?

A. In this example, Pippy moves forward one square, turns left, moves forward one square, turns right, moves forward one square. Repeat twice from “turns left”, and then Pippy gets to the cheese.

B. In this example, the Green Simple Loop is executed twice because of the Number 2 card. The loop is defined as “turns left, moves forward one square, turns right, moves forward one square”.

Hands-on Creativity

Try it: Write a different program for this lesson with the Simple Loop Cards. If Pippy cannot get the cheese, try debugging to find the error.

Model Operation Video

Evaluation

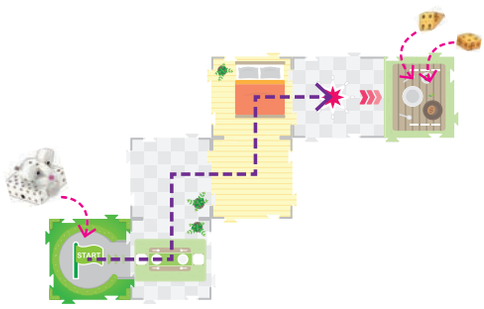
1 Model Assembled

2 Experiment Complete

3 Model Creation

8 Zig, Zag, and Twirl for Cheese

Learning Goal: Sensor



Pippy smells some cheese on the picnic table. It smells strong, so it must be two pieces of cheese! Can you program Pippy to first zig-zag through the house and into the backyard, and then to spin around in a circle when she reaches the cheese?

First, record the main program. Then, lift the robot up and record the function starting with the Red Function Start card. The robot will save both the main program and the function in its memory. Then run the program on the map.

CHECK IT OUT

Sensors are used to detect changes in the environment and send messages to a computer or other electronic device. Sensors can detect external signals like light, heat, humidity, smoke, etc. For example, automatic doors in the super markets use an infrared sensor to measure distance. When an item approaches, the sensor notifies the computer to open the door.

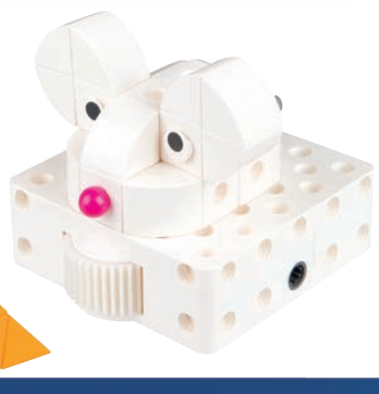
The Robotic Base Units in this package have a sensor. At the bottom of the robot is an optical identification (OID) sensor that scans the invisible patterns on the cards. The microprocessor inside the robot records the program and converts the read data into commands that the robot can execute.

Brainstorming

What equipment in daily life uses sensors?

Pippy & two cheeses 's part list & assembly steps:

Please refer to Lesson 6.



[CODE]

MAIN PROGRAM:

RED FUNCTION:

A.  OR B. 

What's Happening?

The zig-zag program works the same way it did in the previous lesson. But this time, there is a base map card (the card with the red star on it) at the end of the path. When Pippy walks to the red function base map card, the optical identification (OID) sensor senses the invisible pattern on the card and Pippy automatically performs the program.

In order to make Pippy turn around in a circle, please use four Turn Left (counterclockwise) cards in the red function program. That is because one Turn Left card rotates the robot 90 degrees, it needs four Turn Left cards to turn 360 degrees, or use one Turn Left card and the Number 4 card to write the program.

When the program is executed, Pippy moves in a zig-zag pattern to the picnic table to find the cheese, and spins around in a circle.

Hands-on Creativity

Try it: Write a program to let Pippy spin around in a circle clockwise after finding the cheese.
Can you write the correct program?

Model Operation Video

Evaluation

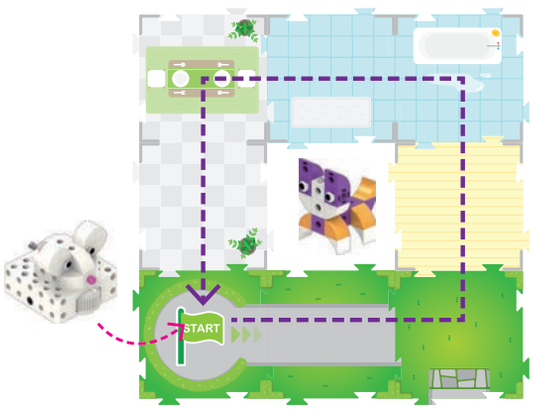
1 Model Assembled

2 Experiment Complete

3 Model Creation

9 Teasing Purry

Learning Goal: Computational thinking



Pippy has lots of energy from all the cheese she has been eating. Now, she wants to play one of her favorite games: running around the house and the neighborhood, running right by Purry, and trying not to get caught.

Can you program Pippy to run in a square route around Purry, and back to the start again? Can you do it with one loop, to use fewer code cards?

CHECK IT OUT

The idea of computational thinking is not to write a program, but to tell the computer what to do.

If we are going to the park today, we may plan the route before going out, and we may think about some routes to get to the park and which road is the best.

For example, the best route may be the shortest, or fastest, or passes by our favorite store. After making the decision, we follow the plan step-by-step.

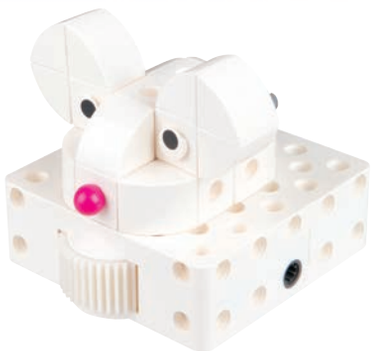
The thinking process of the example above is called computational thinking, which performs actions according to the instructions. It is like programming.

It is useful to break down complex problem into small and simple problems that we can easily understand.

Brainstorming

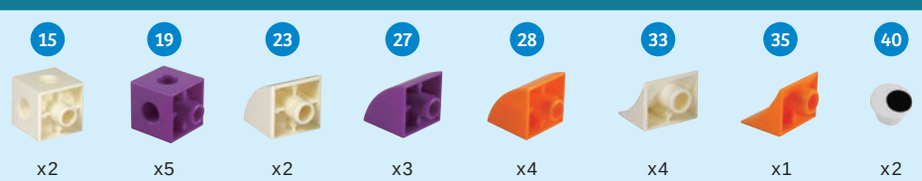
Pick a park near your home and try to plan your route to the park.

Pippy 's part list & assembly steps:

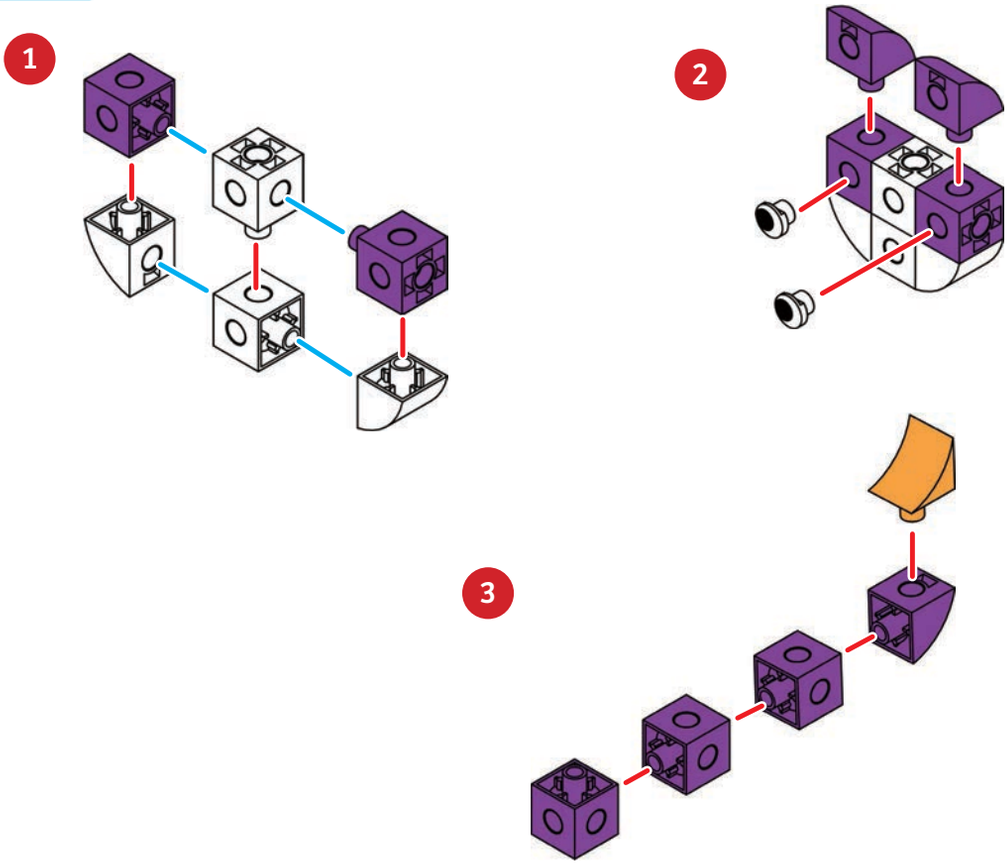


Please refer to Lesson 6.

Parts List

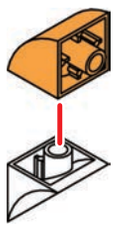


PURRY

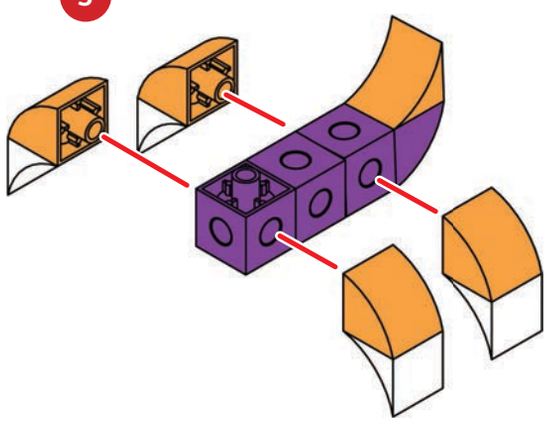


9 Teasing Purry

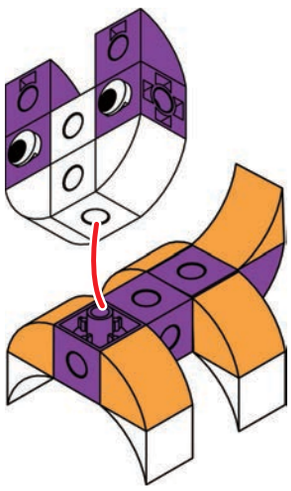
4 ×4



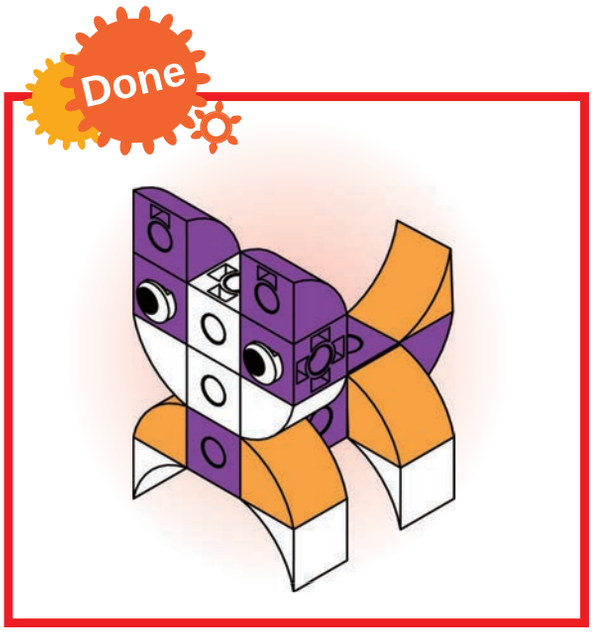
5



6



Done



Smart Manual
Web Service

[CODE]

A.

OR

B.

OR

C.

What's Happening?

Three examples of programs that will complete this lesson are pictured below. Example A uses no loops and is almost twice as long as the other two.
Examples B and C both use one loop in similar ways. The difference between Examples B and C is that Example C uses number cards to repeat the Move Forward commands.
When the program is executed, Pippy walks a set of codes four times: Move Forward two squares, and Turn Left. Pippy completes a counterclockwise route to avoid Purry.

Hands-on Creativity

Try it: Swap the start map card with the table map card at the top left, and put Purry on the same map card. Find the solution and write the shortest program for Pippy to run in a Square route past Purry, and back to the start again.

Model Operation Video

Evaluation

1 Model Assembled

2 Experiment Complete

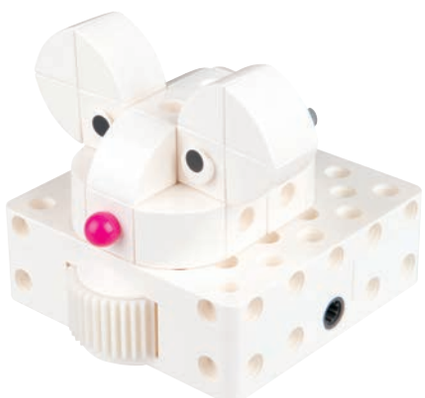
3 Model Creation

10 Monograph 2

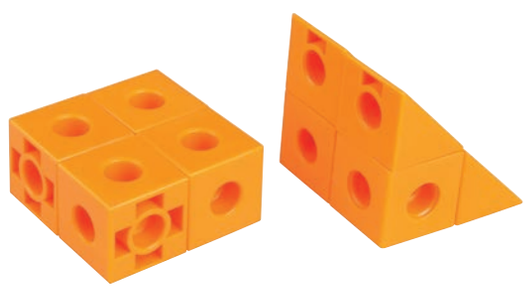
Pippy is hungry now and she wants to eat cheese, but she is afraid of meeting Purry or being chased by Purry. Pippy comes up with a great idea: run in a figure 8 route to avoid meeting Purry and get the cheese.

Place Purry and the cheese on the map at the designated locations. Write a program that lets Pippy run in a figure 8 route to avoid Purry and to get the cheese. After getting the cheese, Pippy returns to the start. Try and write the program with two loops.

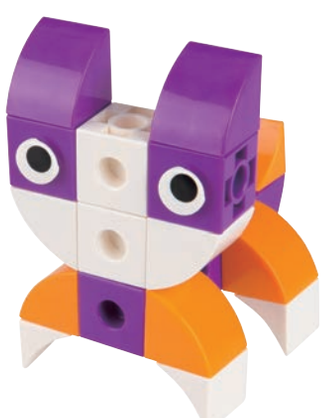
Model Review



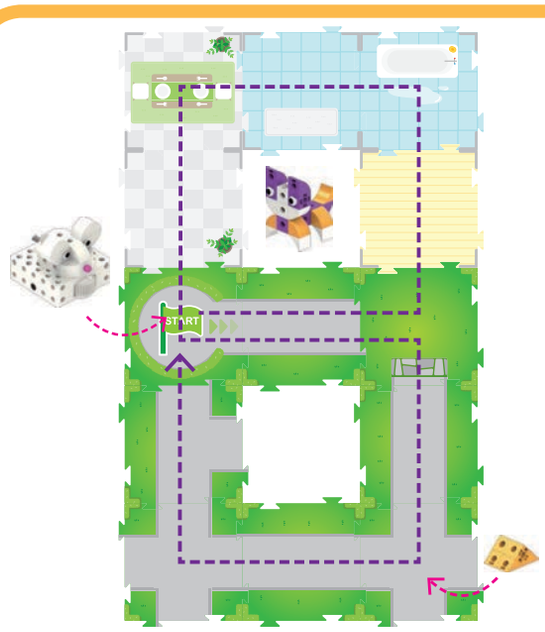
1. Pippy



2. Cheese



3. Purry



4. Map

Design Concept

My Artwork

Evaluation

1 Model Design

2 Model Creation

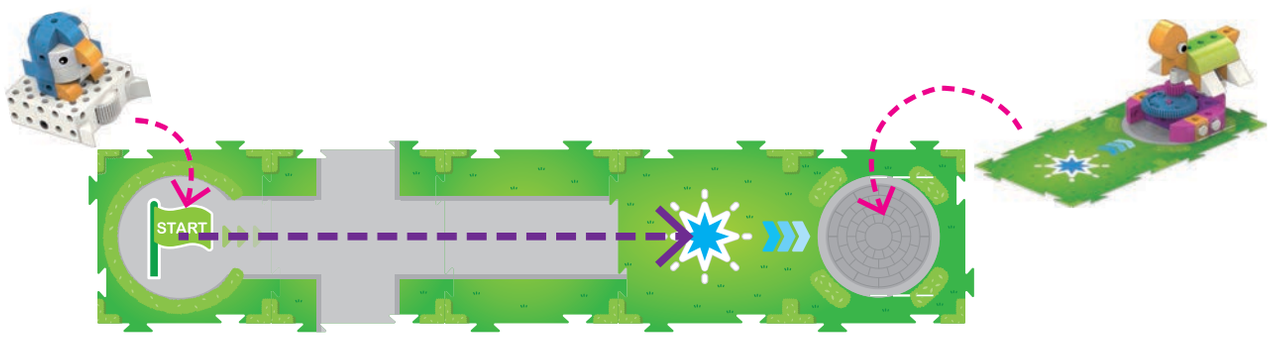
3 Winner!

11 Arty Dances with Tucker

Learning Goal: Power transmission

Arty is a penguin. Arty didn't like the cold weather in Antarctica, so he moved to a pleasant park with grass and trees. Arty lives in the park with his friend Tucker the Turtle.

Arty wants to visit his friend Tucker. Can you write a program to make Arty drive to Tucker, and when he gets there, to perform a function to spin Tucker around in circles?



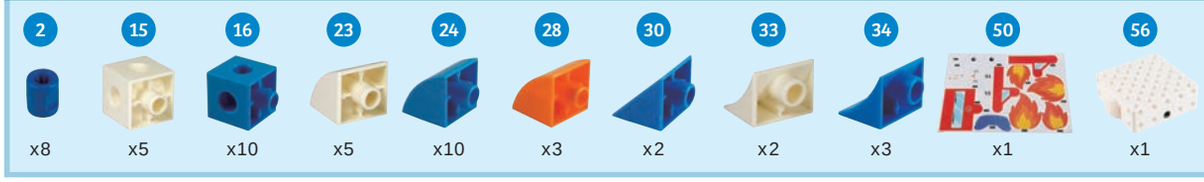
CHECK IT OUT

How does Arty dance with Tucker? Using the benefits of gear power transmission, Arty can dance with Tucker. Arty has an output gear in the robotic base unit. Tucker also has a gear. The two gears mesh with each other so that the motor drives the output gear and transmits power to the 60T GEAR, making Tucker spin around in circles.

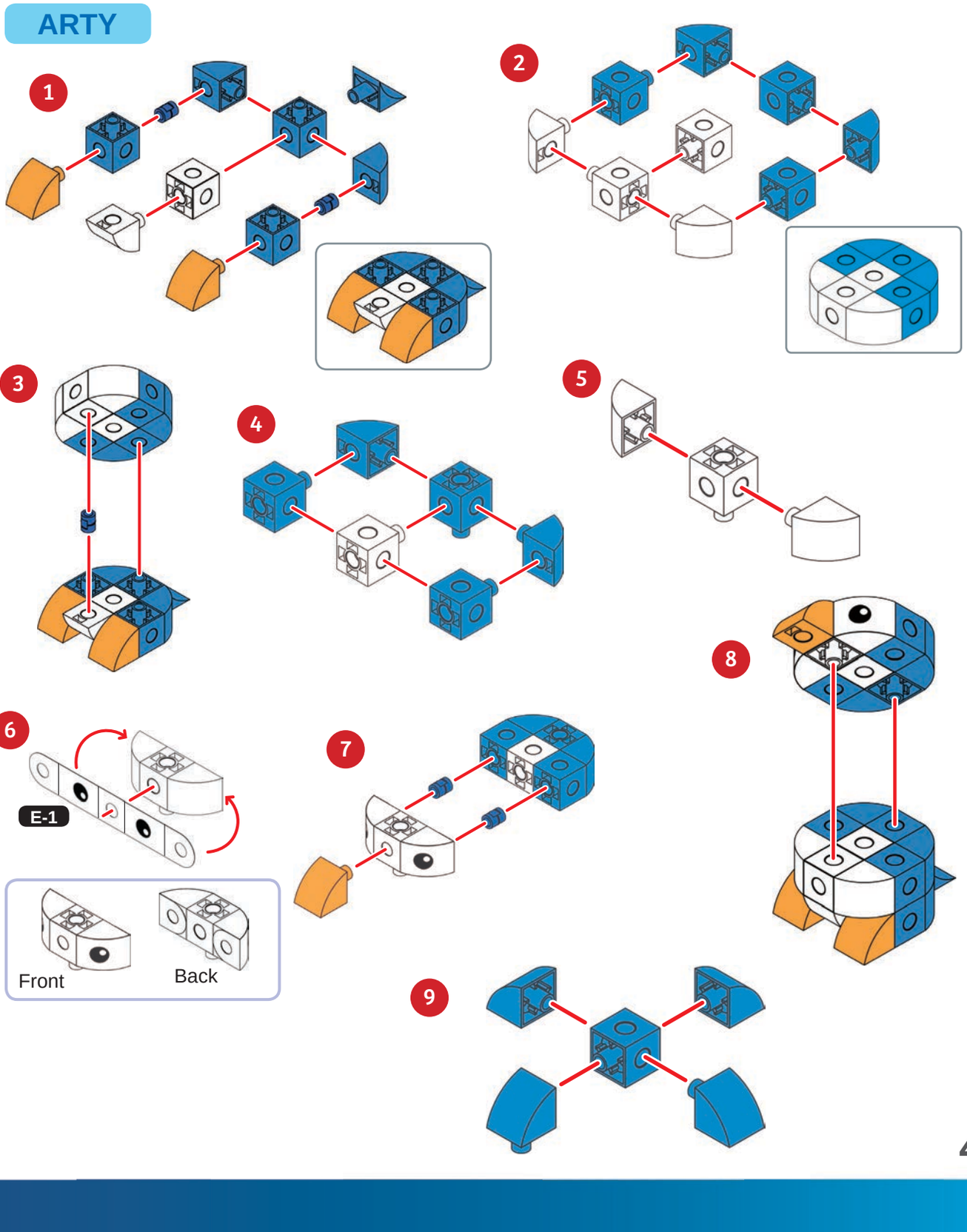
Brainstorming

What kind of code cards do you need to use when writing a program to make Arty dance with Tucker?

Parts List

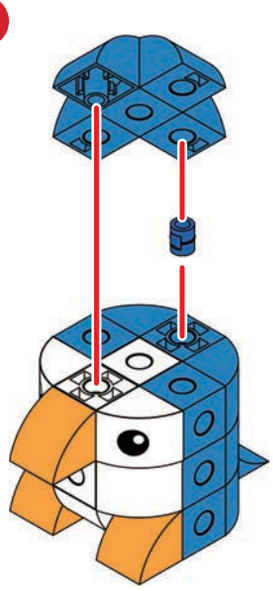


ARTY

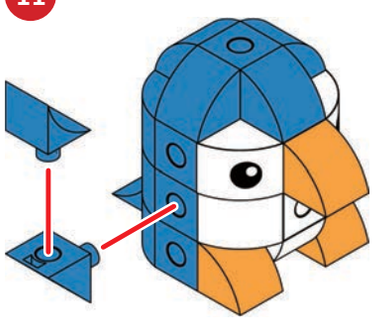


11 Arty Dances with Tucker

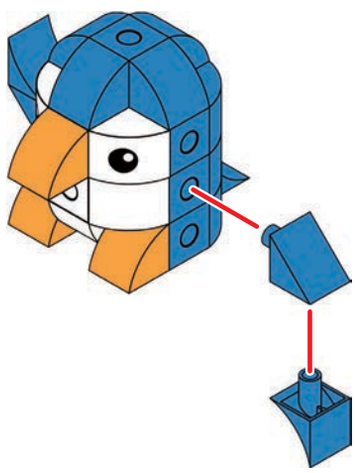
10



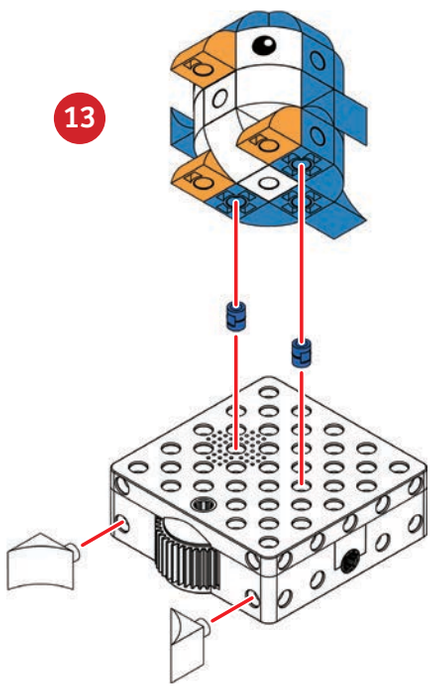
11



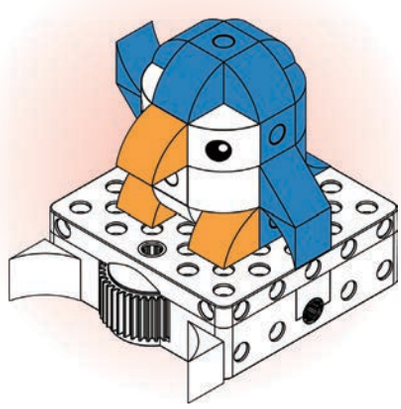
12



13

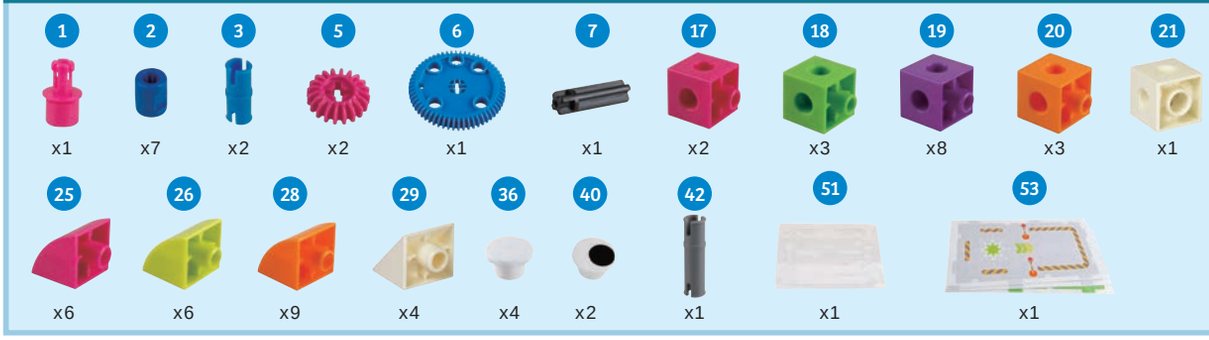


Done

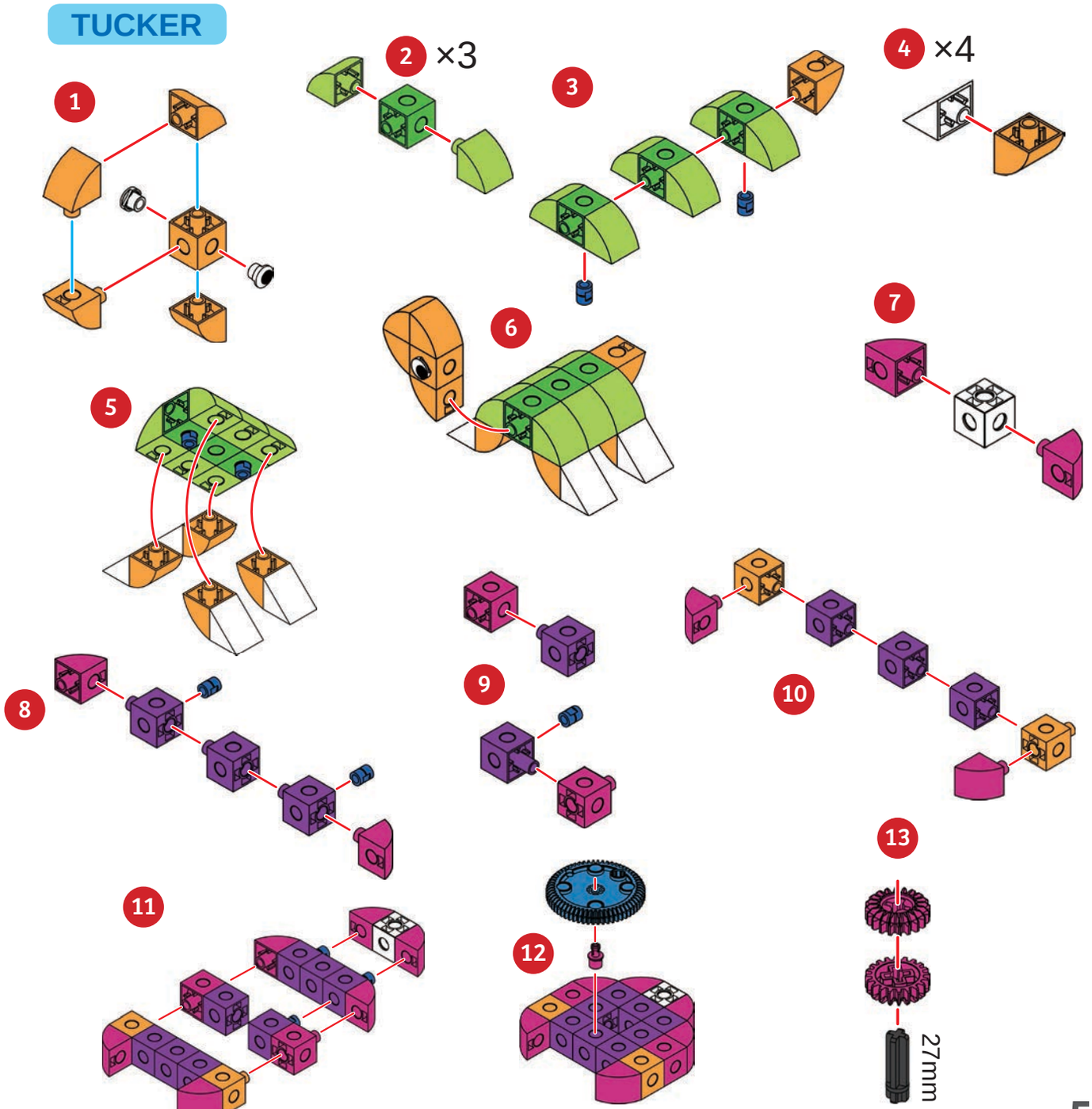


Smart Manual
Web Service

Parts List

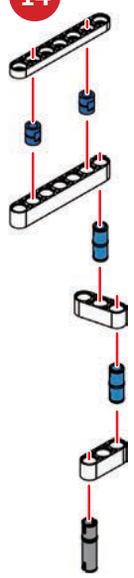


TUCKER

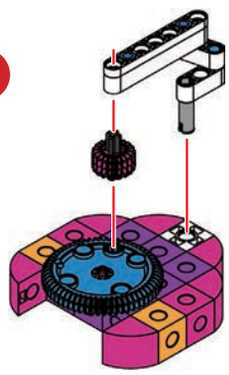


11 Arty Dances with Tucker

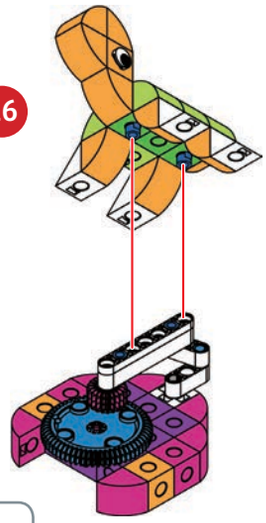
14



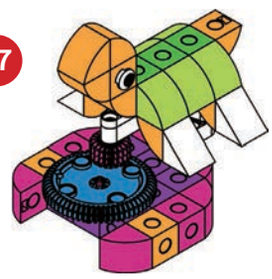
15

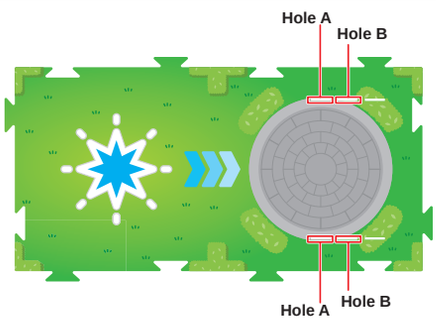


16



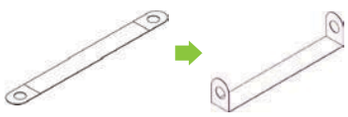
17





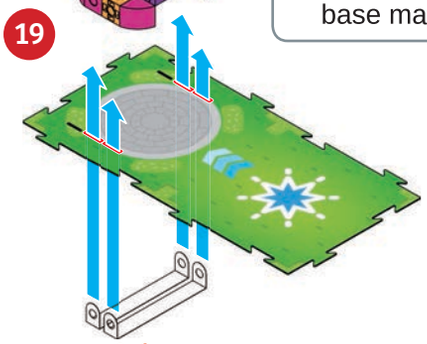
Now attach the turtle to this base map card.

18 ×2

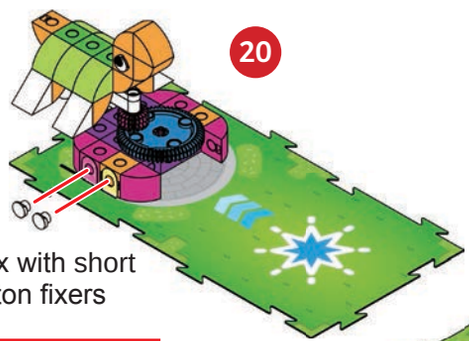


Plastic map card straps

19

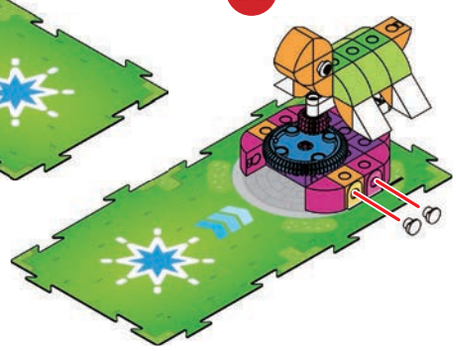


20

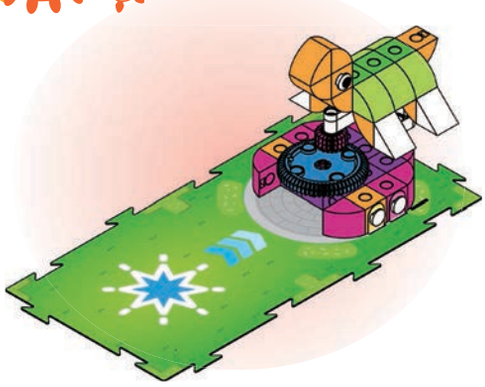


Affix with short button fixers

21



Done



Smart Manual Web Service

[CODE]

MAIN PROGRAM:

BLUE FUNCTION:

What's Happening?

The main program brings Arty to the base map card with the blue star on it.

The program of the blue function turns the output gear right (clockwise) and then turns the output gear left (counterclockwise).

When the robot scans the base map card, the robot automatically moves into position and runs the Blue Function Code, which instructs the robot to turn the output gear first clockwise and then counterclockwise. The gear meshes with the gear connected to Tucker, so this makes Tucker turn around as well.

Hands-on Creativity

Try it: Change the program of the blue function in this lesson. Can you let Arty and Tucker dance for a longer period of time?

Evaluation

Model Operation Video

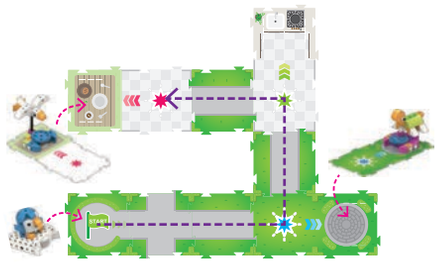
1 Model Assembled

2 Experiment Complete

3 Model Creation

12 Commotion in the Park

Learning Goal: Function



Arty visits Tucker, who dances around because he's so happy to see Arty. Then Arty goes to his picnic table, and sees that Gully is trying to grab his fish dinner! He does an elaborate dance to shoo Gully away from his food.

Write a program to let Arty visit Tucker and dance together, and then Arty goes to the table to shoo Gully away with the red function.

CHECK IT OUT

A more complicated program will often have many program commands that need to be executed repeatedly. If you singularly add these program commands, the program will be too long. The solution is to write a set of program commands with a specific function as a unit. This kind of programming is called a "function" in the programming language.

In daily life, you can find the application of this function in a washing machine. Each time you wash the clothes using a washing machine, you need to press the water level, washing process, times of rinse, spin and other buttons on the control panel for your desired washing needs. Some washing machines have a one touch automatic button. When you press that button, the laundry program is automatically chosen and executed. This one touch button is a “function” in programming.

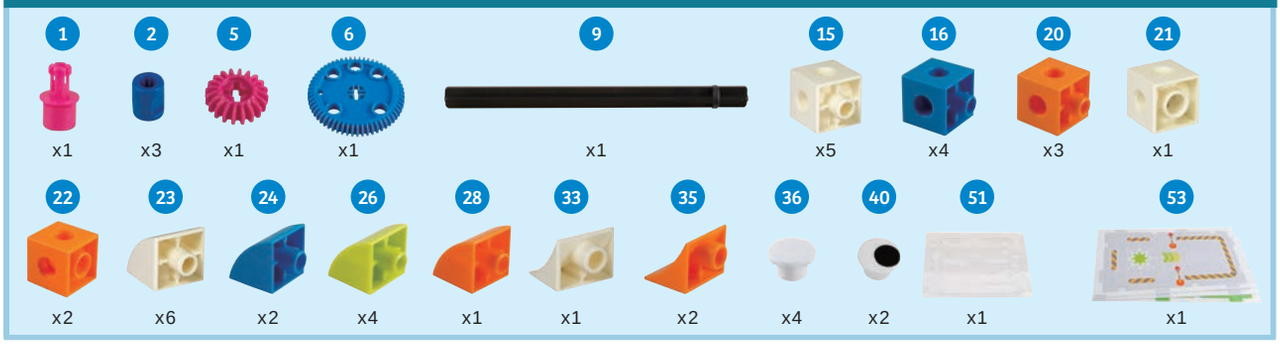
Brainstorming

Is there anything in daily life that can be done repeatedly day after day? Write these things that are repeated every day. How would you program them? (For example: Dad's one-day routine as a function: get up, have breakfast, work, have lunch, work, have dinner, rest, sleep.)

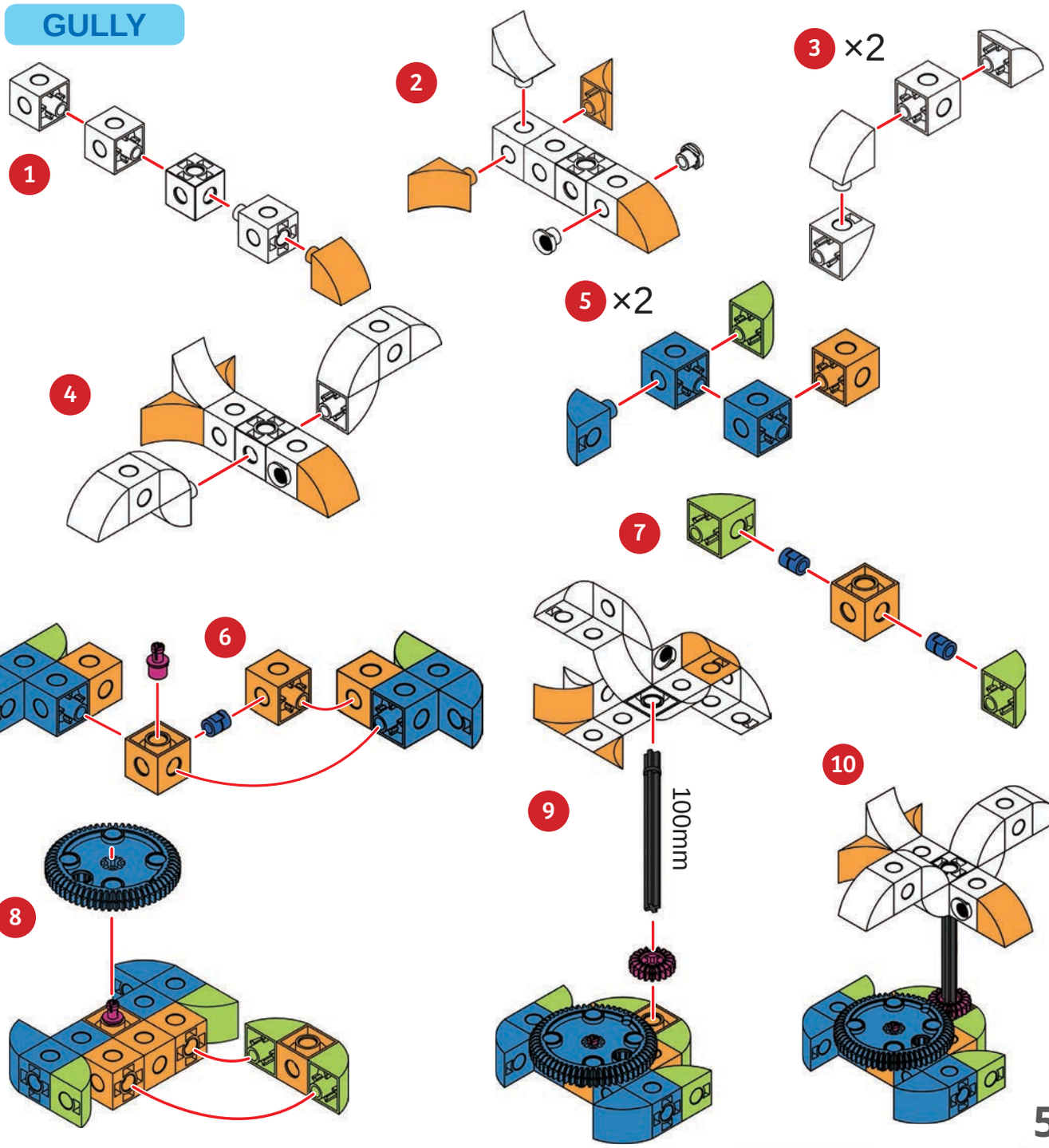
Arty & Tucker 's part list & assembly steps:

Please refer to Lesson 11.

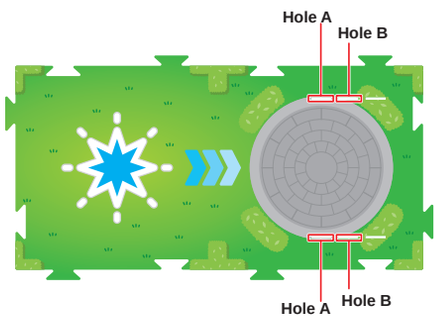
Parts List



GULLY

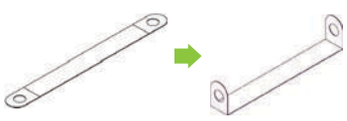


12 Commotion in the Park

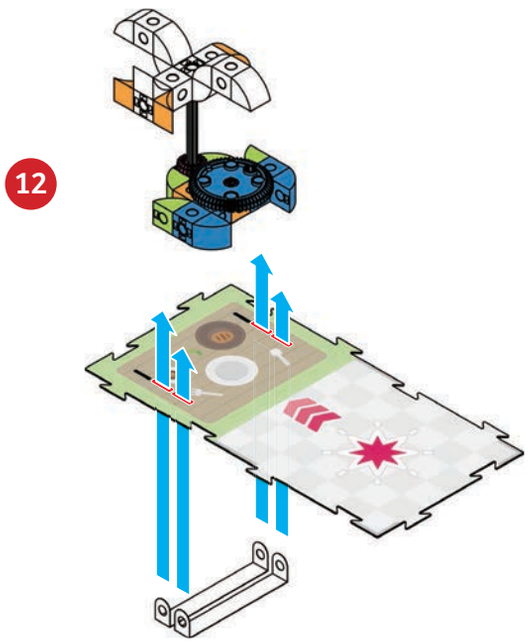


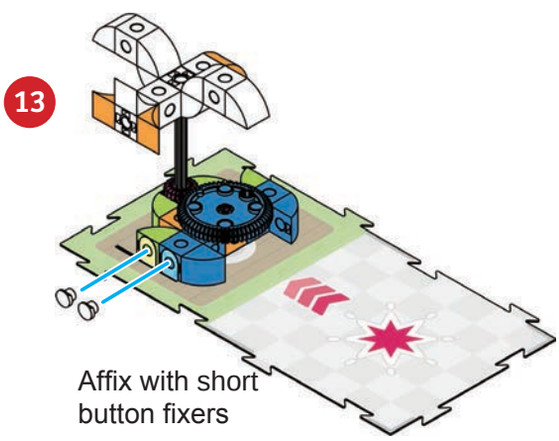
Now attach the turtle to this base map card.

11 ×2

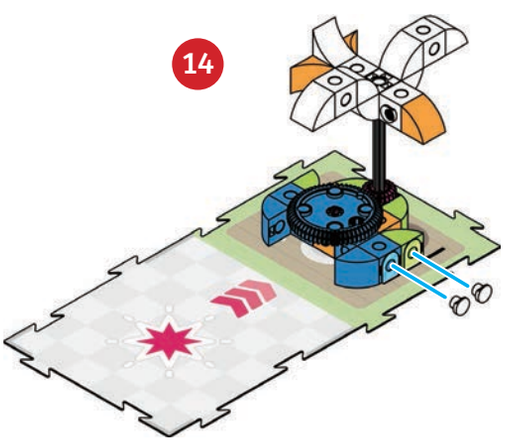


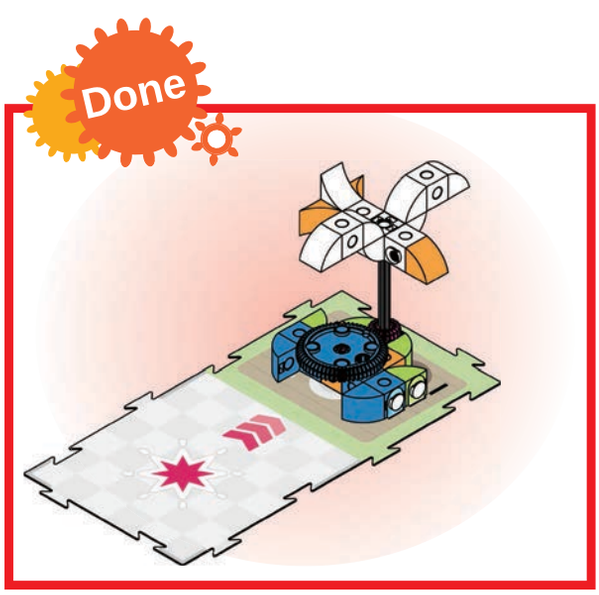
Plastic map card straps





Affix with short button fixers



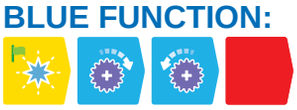


Smart Manual Web Service

[CODE]

MAIN PROGRAM:

BLUE FUNCTION:



RED FUNCTION:

What's Happening?

The main program brings Arty to Tucker, who spins around just like in the previous lesson. Then, the main program moves Arty along to the base map card with the red star on it.
The program of the red function lets the output gear turn right (clockwise) 4 times, pauses for one second, then the output gear turns left (counterclockwise) 4 times. The program repeats between the loop cards once, so the program runs twice. Here, the Red Function causes Arty to perform a clockwise and counterclockwise spinning action two times. This causes Gully to spin around, as if to fly away from Arty's dinner.

Hands-on Creativity

Try it: Using only the Turn Right (clockwise) card, can you program the path from the start to the picnic table?

Model Operation Video

Evaluation

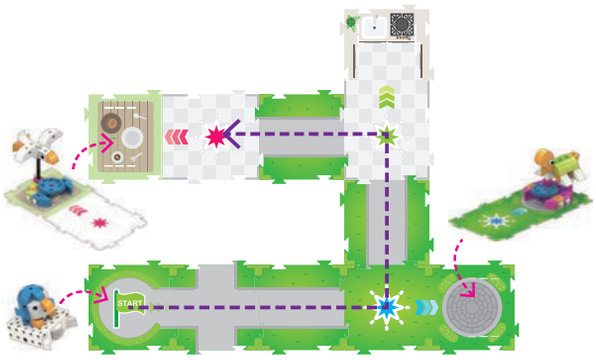
1 Model Assembled

2 Experiment Complete

3 Model Creation

13 Arty Starts Squawking

Learning Goal: Subroutine



Arty visits Tucker again, but this time Arty talks to Tucker when he sees him. Do you recognize part of Lesson 12 that can be repeated in a simple loop? Repeat Lesson 12, but this time try to use a simple loop in the main program. Also, try to add a sound code card to the Blue Function.

CHECK IT OUT

A subroutine has the structure of a complete program. The format, input, output, operator, loop, judgment, etc., in the program can be used in the subroutine. Also a subroutine can be in another subroutine.

A subroutine is usually used in the following two situations:
(1) In order to make the structure of the program clear, we separate a special part of the function; this part is called a subroutine, which is different from the main program.
(2) Some programs (such as drawing or mathematical functions) can be used repeatedly by the same or different programs. We would like to write this kind of usable program as a subroutine, making it simple and convenient, it also prevents errors.

Brainstorming

How many subroutines are there in this lesson's program? Please point them out.

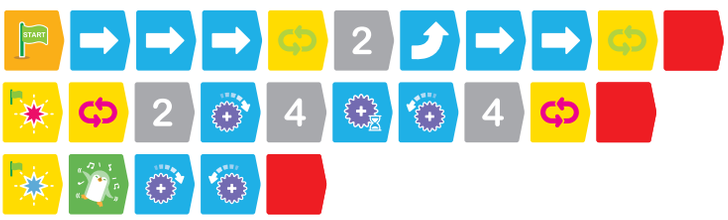
Arty & Tucker 's part list & assembly steps:

Please refer to Lesson 11.

Gully 's part list & assembly steps:

Please refer to Lesson 12.

[CODE]

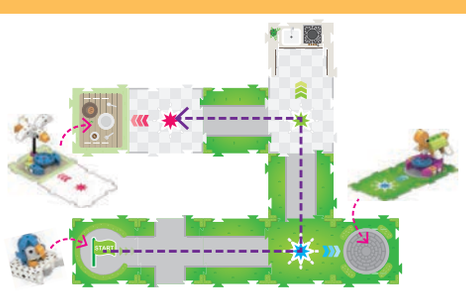


What's Happening?

The main program moves Arty three squares to Tucker, and then repeats the set of following codes: “turn left, move forward two squares” to move Arty to the Red Function base map card.
There are two subroutines in this lesson: Blue Function and Red Function.
The Blue Function instructs Arty to turn the output gear like the previous lesson, but adds a Penguin sound card. When Arty walks to the Red Function base map card, Arty plays a penguin sound, and then turns the output gear right and turns the output gear left.
The Red Function instructs Arty to turn the output gear right (clockwise) 4 times and then pause for one second and turn the output gear left (counterclockwise) 4 times. The program repeats the loop cards once, which results in running the program twice.

Hands-on Creativity

Try it: Add a green function base map card as shown, and make a tree model on the green function base map card. Write a green function subroutine to let Arty walk to the tree and take a rest for 5 seconds and play an "Ahh!" sound after Arty visits Tucker, and then go to the picnic table and shoo away Gully.



Evaluation

Model Operation Video

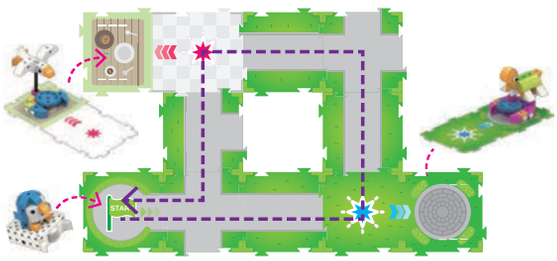
1 Model Assembled

2 Experiment Complete

3 Model Creation

14 Arty's Complete Trip

Learning Goal: Mechatronics



This time, Arty wants to go back to his starting point after he gets Gully to leave his food alone. Repeat Lesson 13, but this time add a map card that allows Arty to get from the base map card back to the start. Edit the main program to bring Arty back to the start. And also, add some more penguin sounds to the interaction Arty has with Gully.

CHECK IT OUT

In this package, children can understand the concept of electromechanical integration with programming and machines. For example, one program makes Arty do actions as the sensor detects a function on the base map card, resulting in Arty and Tucker dancing together by meshing their gears. In daily life, we can often see different uses of electromechanical integration. Take the air conditioner as an example. When we set a temperature, the air conditioner will receive the indoor temperature through an electronic sensor. If the indoor temperature is too high, the mechanical operation in the air conditioner will cool down the temperature. If the temperature is too low, the machine stops running. Electromechanical integration is the integration of the mechanical structure with power supply (electricity) to make the function work.

Brainstorming

Can you give another example of electromechanical integration in life?

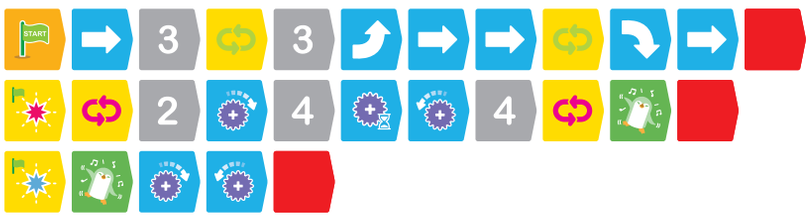
Arty & Tucker 's part list & assembly steps:

Please refer to Lesson 11.

Gully 's part list & assembly steps:

Please refer to Lesson 12.

[CODE]



What's Happening?

The main program moves Arty three squares to Tucker, and then Arty repeats this set of following codes three times: “turn left, move forward two squares” (using loop cards). This allows Arty to turn right and move one square forward to the Start.

The blue function is the same as the previous lesson. The red function instructs Arty to turn the output gear right (clockwise) 4 times and then pause for one second and turn the output gear left (counterclockwise) 4 times. The program repeats between the loop cards once, so the program runs twice. A Penguin sound card makes Arty shoo away Gully.

Hands-on Creativity

Try it: Add a new light effect to the blue function using the Falling Star card, and observe the result of the experiment.

.....

.....

Evaluation

Model Operation Video

1 Model Assembled

2 Experiment Complete

3 Model Creation

15 Monograph 3

Arty's good friend, Gigi, moves to the park. Arty wants to visit Gigi's new home. First, Arty will pass Tucker's house. Arty decides to visit Tucker before visiting Gigi. It is a wonderful day with friends.

Please make 1 or 2 Gigi fish. Place Tuck on the blue function base map card, and place Gigi fish on the green function base map card.

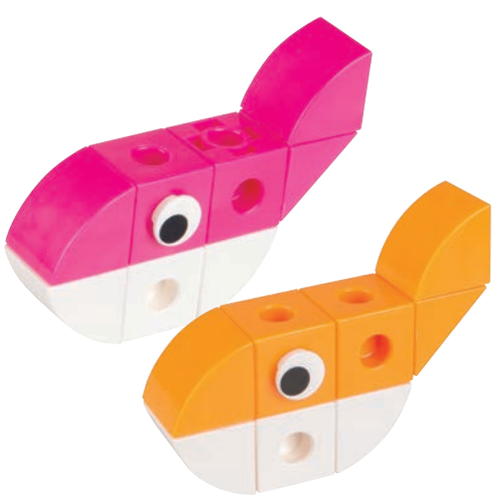
Write a program to let Arty start from the starting point to visit Tucker and dance with Tucker, and then meet Gigi saying "Hi!" with Rainbow Light effect twice, and finally return to the starting point.

(Note: Write a main program route, and two subroutines that interact with Tucker and Gigi.)

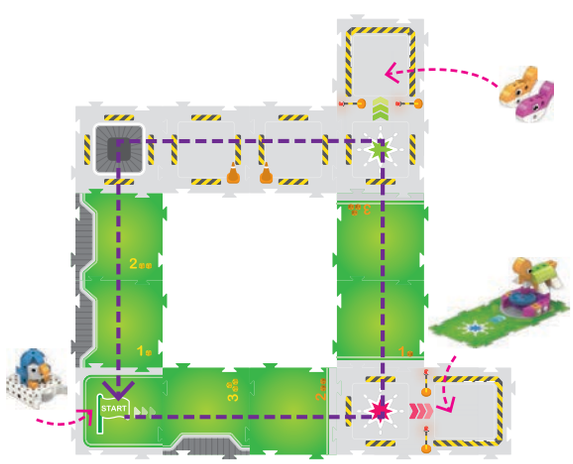
Model Review

1. Arty

2. Tucker



3. Gigi (Fish)



4. Map

Design Concept

My Artwork

Evaluation

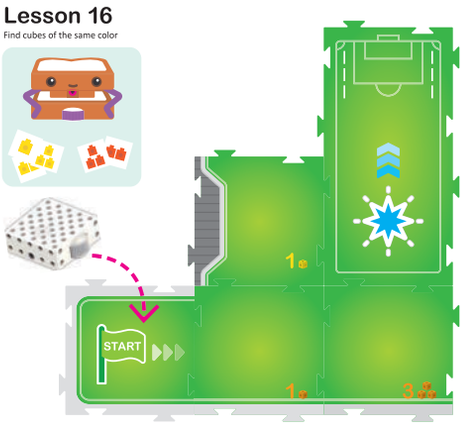
1 Model Design

2 Model Creation

3 Winner!

16 Find Cubes of the Same Color

Learning Goal: Classifying



Scan the “Lesson 16”code graphic on page 73. Program the robot to drive only on map cards with numbers of the same color printed on them, and end up on the blue star.

CHECK IT OUT

The meaning of classification is to divide things into groups according to their type or attributes or similarities. For example, there is a pile of building blocks and we can divide them into different groups according to their attributes - color, shape, function.

In the process of classification, children can recognize and distinguish the similar characteristics of things, and then determine whether they belong to the same group. The ability of classification is the basis for learning concepts and definitions and for understanding counting a collection.

Brainstorming

How did the store divide the clothes on the shelves in the following picture?

Parts List

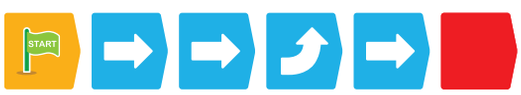
x1

x1

x1

[CODE]

Finding orange numbers:



Hands-on Creativity

Try it: Rearrange the map cards from this lesson, and then write a program again to find cubes of the same color based on the new map.

Evaluation

Model Operation Video

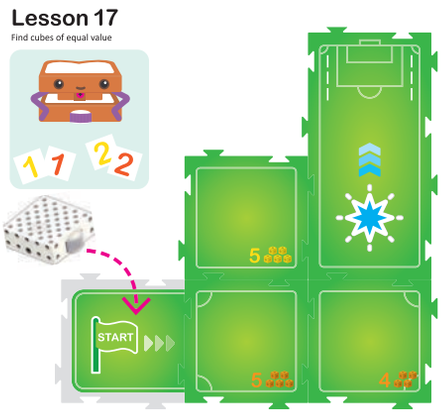
1 Model Assembled

2 Experiment Complete

3 Model Creation

17 Find Cubes of Equal Value

Learning Goal: Matching pair



Scan the “Lesson 17”code graphic on page 73. Program the robot to drive only on map cards with numbers of the same color printed on them, and end up on the blue star.

Coding Concepts

Pairing is to match objects in pairs, that is, to choose two identical objects in pairs. With the game of pairing, children can understand the meaning of "identical" and learn the concept of numbers. When objects of the same characteristic are paired, it means that the two objects have the same content or equal value.

It is important to identify the image of numbers for children. If children can match the abstract numbers with the physical objects, or the represented pictures, or the oral numbers they speak, parents can tell that these children truly understand the concept of numbers.

Brainstorming

Please find the same pattern of socks and match them together by drawing lines.

Parts List

x1

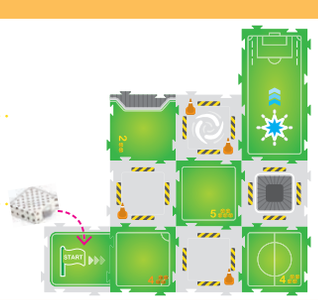
x1

x1

[CODE]

Hands-on Creativity

Try it. Make the map cards as shown, and then write a program to find cubes with the same value (or amount) printed on the map cards.



Evaluation

Model Operation Video

1 Model Assembled

2 Experiment Complete

3 Model Creation

18 Find Cubes in Sequence

Learning Goal Sequence (Mathematical concept)

Lesson 18 & 19
Find cubes in sequence


Increasing value


Decreasing value

Scan the "Increasing Value" code graphic on page 73. Program the robot to drive on map cards with numbers of increasing value printed on them, ending at the blue star.

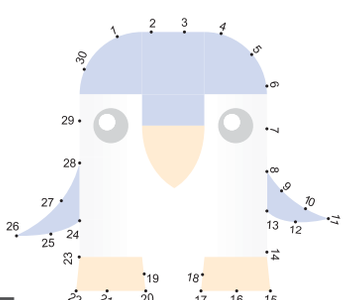
Coding Concepts

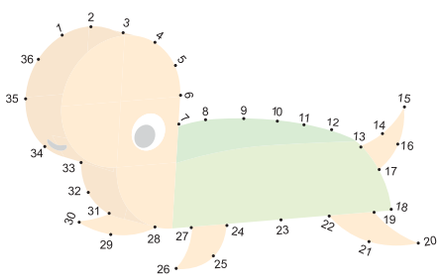
The ability to observe, analyze, and judge is called "observation" and "comparison". “Sequence” is to find a pattern or rule that makes an association amongst items.

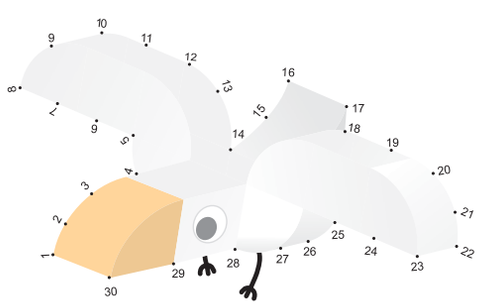
There are different types of sequences, including high-short sequences, fat-thin sequences, size sequences, amount sequences, weight sequences, color sequences (from darkest to brightest), time sequences, and regular sequences.

Brainstorming

Please connect numbers in increasing value to make a pattern by drawing a line. What is the hidden pattern? Hurry up and connect the numbers!







Parts List

x1

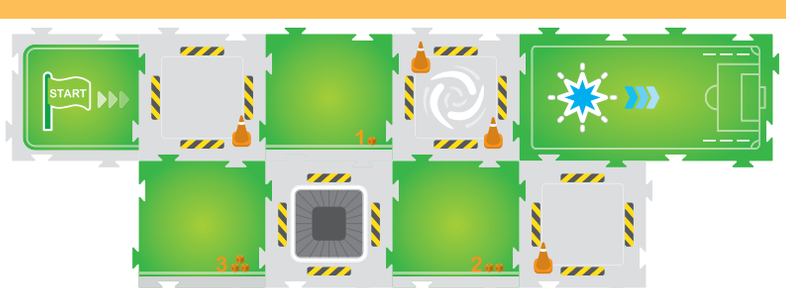
x1

x1

[CODE]

Hands-on Creativity

Try it. Please make the map as shown. Scan the "Decreasing Value" code graphic on page 73. Please write a program that allows the robot to drive on map cards with numbers of decreasing value printed on them, ending at the blue function base map card.



Evaluation

Model Operation Video

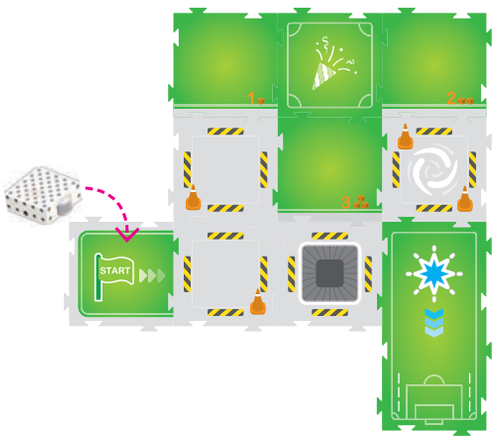
1 Model Assembled

2 Experiment Complete

3 Model Creation

19 Find Cubes in Sequence

Learning Goal: Sorting



Scan the "Decreasing Value" code graphic on page 73. In a program, program the robot to drive on map cards with numbers of decreasing value printed on them, ending at the blue star.

Coding Concepts

Sorting is an important logical ability in the cognitive development of younger children. The thinking processes included reversibility, transitivity and seriation. Children can compare the differences.

Seriation in sorting refers to the fact that any one of the elements in the sequence of objects arranged by the equidistant relationship is larger than the previous one and smaller than the latter. These abilities use abstraction, generalization, and reasoning of thinking. Therefore, when children actually master these three thinking process, children also know how to think abstractly, generally, and with reason.

You can train your child how to sort any object, by practicing positive and negative sorting of numbers or positive and negative sorting of amounts. For example, sort by quantity: from more objects to fewer objects (or from fewer objects to more objects).

Brainstorming

Observe these three glasses of juice. Which glass has the most juice? Which glass has the least juice? Please point out the three glasses of juice from the least amount of juice to the most.

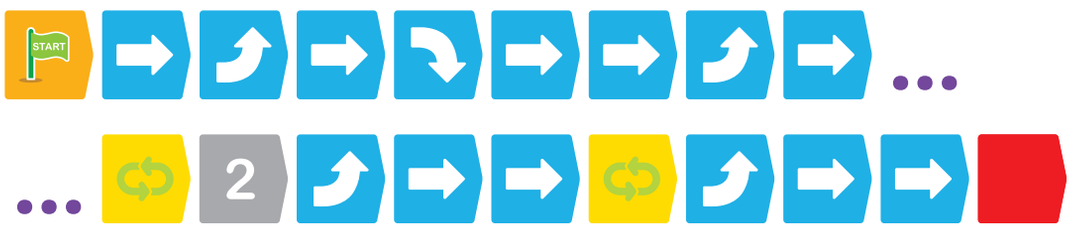
Parts List

x1

x1

x1

[CODE]



(Decreasing value)

Try it. Use the map of this lesson. This time, please write a program to make the robot drive on map cards with numbers of increasing value. For example: go from the starting point, and then go to 1, 2, 3, and finally go to the blue function base map card.

Hands-on Creativity

Model Operation Video

Evaluation

1 Model Assembled

2 Experiment Complete

3 Model Creation

20 Monograph 4

Based on previous lessons, please draw lots to determine a number you are looking for. Program the robot to drive only on map cards with numbers of the same value (or amount) printed on them, and end up on the red function base map card.

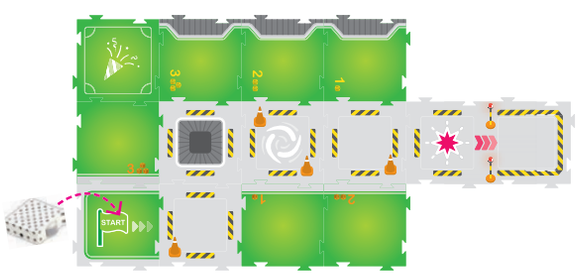
Before the robot runs the program, please scan the "Find Cubes of Equal Value" code graphic on page 73.

Model Review

1. Robotic Base Unit

2. Find Cubes of Equal Value

3. Draw Straws



4. Map

Design Concept

My Artwork

Evaluation

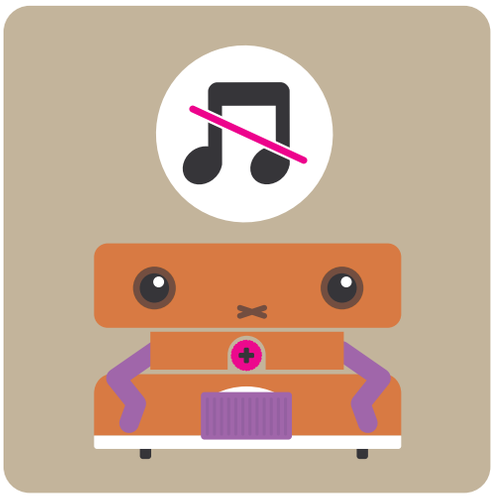
1 Model Design

2 Model Creation

3 Winner!

Background Music

ON

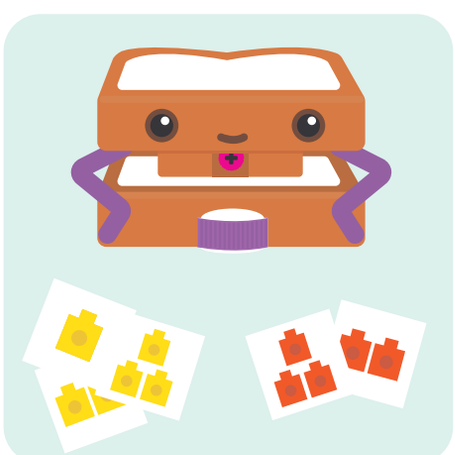


OFF

Math Programs

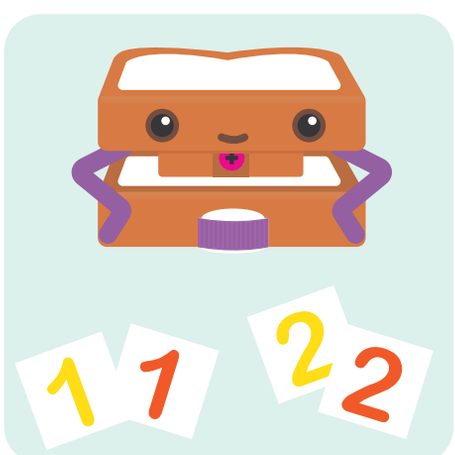
Lesson 16

Find cubes of the same color



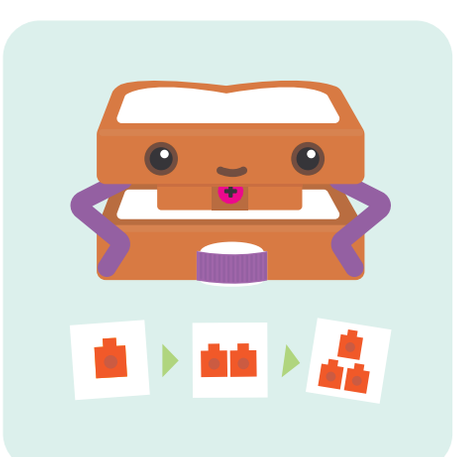
Lesson 17

Find cubes of equal value

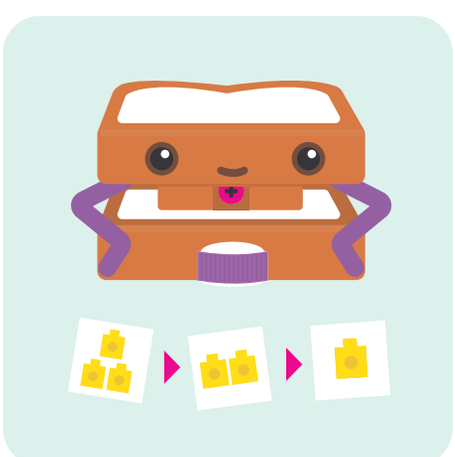


Lesson 18 & 19

Find cubes in sequence



Increasing value



Decreasing value

MADE IN TAIWAN

GENIUS TOY TAIWAN CO., LTD.

www.gigotoys.com

 R21#1276A