

Functions

Overview

Concept	Function —— A group of coding instructions that can be called over and over again
Explanations	Example: Wash hair
Learning Objectives	1. Understand the concept of Functions; 2. Create functions and call the functions to perform tasks.
Teaching Preparation	1. Paper strips of functions (step 3 - Lead-in game) ; 2. A whiteboard and a whiteboard marker (or you can use a blackboard and chalks); 3. One Codey Rocky and a Bluetooth dongle (or a USB cable) per student but it's fine if 2 or 3 students share one set; 4. A computer per student, but it's fine if 2 or 3 students share a computer. And install mBlock 5 on each computer; 5. A self-review report form per student.
Time	120-240 min

Teaching Procedure

Step 1: Review——Conditionals

Review:

- What is a conditional block?
- How did we use the conditional blocks to assess conditions in the previous lesson? Ask

students if they could come up with any of those conditions.

The conditional blocks assess various conditions and decide whether the conditional statements are true or not. If the conditional statement is true, then the computer will run the piece of code. Otherwise, it will skip it or ignore it. In the previous lesson, we used the conditional blocks to assess the following situations: 1) whether the color detected by the color sensor is green 2) whether there are obstacles ahead 3) whether the light intensity exceeds 20 4) whether the sound loudness is less than 20 5) whether the reflected infrared light intensity exceeds 10.

Step 2: Explain New Knowledge —— Functions

Today, we are to have a look at **Functions**. In daily life, we often do some things more than once, like washing hair. In most cases, the three steps to wash our hair include: take some shampoo, massage the hair to make the shampoo foam, and rinse the hair with warm water. One of your friends comes to ask you out, but you have to wash hair first. Then you might say, “I’m about to take shampoo, massage the hair to make it foam, and rinse the hair. Allow me some time.” Mom is shouting the breakfast is ready, while you are washing your hair. You then repeat the words, “I’m taking shampoo, massaging the hair to make it foam, and rinsing the hair.” Being caught in the rain, you are murmuring to yourself, “I’ve got to take shampoo, massage my hair to make it foam and rinse the hair.”

Actually, we don’t describe each step in everyday life, instead of which we often give the group of steps one simple name. We use the name to represent the whole sequence of steps and call the name when referring to the steps. For example, we name the sequence of actions above as *wash hair* . When your friend comes to ask you out, you say this time, “I am about to wash hair. Allow me some time. “When your mom is shouting the breakfast is ready, you say, “I’m washing hair, wait.” Being wet in the rain, you murmur to yourself, “I’ve got to wash hair.” Using a simple name to refer to a sequence of steps makes conversations efficient and enjoyable.

In computer programming, we use a function to name a piece of commands and can call the name to run the commands if necessary at any time. However, before that, we have to create a function. The first step is to consider a proper name for a function. Then, we need to input commands under the name of the function to define the function. Here, one thing we must keep in mind is that the function name should be easy to understand, which otherwise could lead to some trouble when we call the function. Take washing hair as an example, if we give the sequence of steps a function name *have dinner* , it could be misleading. How does this sound? When our friends come to ask us out, we tell them “I’m having dinner” while we are washing hair in the bathroom instead. It could be confusing.

After defining the function, we can call the function to run a unit of commands that are stored in the function. For instance, wash hair and then go outside; wash hair and then have breakfast; wet in the rain and then wash hair. Thanks to functions, our programs become neat and easy to understand. We no longer need to repeat the same part of code in programs multiple times.

Step 3: Lead-in Game—— The Morning Functions

Game Steps and Teaching Preparation:

1. Teachers print the first chart appendix in the document, **The Morning Functions**. Cut the print into pieces along the dash lines. These are called paper strips of functions;
2. Teachers put down on the whiteboard “Event -When the alarm clock goes off”;
3. Each student pick one paper strip randomly from teachers;
4. Teachers articulate the game rules: “Everyone has a piece of paper strip in hand, and each of them represents a group of actions. Give the group of actions a name yourself; I will later call the functions under the event of “When the alarm clock goes off”; When the function of yours is called, you need to read aloud the detailed steps and act them out;
5. Teachers write down the functions names on the blackboard;
6. After putting down all the function names on the whiteboard, teachers need to ask students what the order is for the functions to be called. Then write down the function names under the event of “When the alarm clock goes off” in the order;
7. Ask students to think about the functions that are not called or called multiple times. Figure out the reasons and complete the programs.

The following chart demonstrates sample function names:

<u>Oversleep</u> cover head with the quilt; wait for 5 seconds; uncover the quilt; stretch out one hand; turn off the alarm clock.	<u>Have a look at the time</u> take up the alarm clock; have a look at the time; put down the clock.	<u>Get up</u> stretch yourself; uncover the quilt; sit up.
<u>Put on slippers</u> put on the slipper to the left foot; put on the slipper to the right foot; stand up.	<u>Walk</u> lift the left foot; stretch forward the left foot and step on the ground; lift the right foot; stretch forward the right foot and step on the ground.	<u>Brush teeth</u> take the toothbrush; squeeze the toothpaste; rinse the mouth; brush the teeth all around; rinse the mouth.
<u>Wash up</u> turn on the tap; cup the water with hands; wash up with the water; turn off the tap;	<u>Take off the coat</u> If it’s a T-shirt, take it off from top to bottom; If the coat has buttons, unbutton it first, then the left	<u>Take off pants</u> pull down the pants; the left leg comes out of the pants;

dry your face with the towel	arm comes out of the left sleeve, next the right arm comes out of the right sleeve.	the right leg comes out of the pants.
<u>Take clothes off</u> take off the coat; take off pants.	<u>Put on the coat</u> get the left arm into the left sleeve; get the right arm into the right sleeve; if it's a T-shirt, get your head into the shirt; if the coat has buttons, fasten the buttons.	<u>Put on pants</u> get the left leg into the pants; get the right leg into the pants.
<u>Dress up</u> put on the pants; put on the coat.	<u>Carry schoolbag</u> get the right arm through one strap; get the left arm through the other strap; shoulder the schoolbag.	<u>Lace the shoes</u> make a circle of the shoelace with the left hand; make a circle of the shoelace with the right hand; tie the two circles
<u>Put on shoes</u> get the left feet into the left shoe; get the right feet into the right shoe; lace the left shoe.		

Suggestions:

1. The functions of “**Take clothes off**”, “**Put on clothes**”, and “**Put on shoes**” include nested functions, so it might be a little bit challenging for students to understand the concept of nested functions. In consideration of this, we could skip the nested functions. That is to say, we only use the function paper strip “**Dress up**” in the game, ignoring the paper strips of “**Put on the coat**” and “**Put on pants**”. However, if students are old enough to understand the concept of nested functions, then it is fine to introduce the concept to students. A nested function is a function that is defined within another function;

2. In some cases, the function names might be inaccurate. For example, students name the group of actions “lift the left foot; stretch forward the left foot and step on ground; lift the right foot and step on ground” as “Gymnastics”. Have students read the steps aloud, act them out and rename it with the help of partners. Next, try to call the function again in the program.

3. Sometimes, the function names can be funny. For example, give the group of actions “cover head with the quilt; wait for 5 seconds; uncover the quilt; stretch out one hand; turn off the alarm clock” an amusing name like “Annoying”. Don’t limit the name to be “Oversleep”. Rather, encourage students to bring whimsy into the class;

4. If students are not willing to act out the steps or the time is limited, skip the acting out part. Just let them read out the steps.

Game Wrap-up:

Three steps to use a function in a program:

1. Create a function and name it;
2. Define the function;
3. Call the function in a program.

Step 4: Tasks

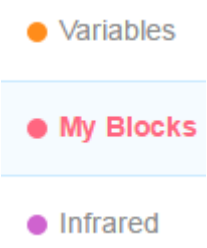
Students work on the following tasks in pairs or on their own. Tell them to tick the square box when they finish the step (The Tasks Cards are included at the end of this document. Print them).

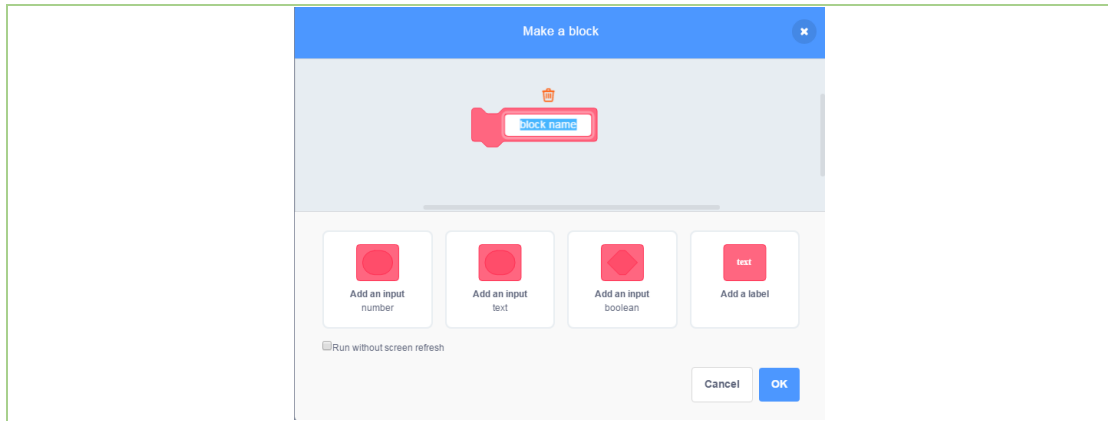
Guided by Teachers: Open the mBlock 5, click **My Blocks**, and select **Make a Block**.

Navigate students to accomplish the task 1.

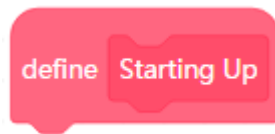
Task 1: Starting up Function

Create a starting up function for Codey Rocky. Each time the Codey Rocky is powered on, the function will be executed.

Task 1-Starting up Function	
Create a starting up function for Codey Rocky	
<input type="checkbox"/>	Click My Blocks at the category bar and select Make a Block . 
<input type="checkbox"/>	Click Make a Block . Create a function and give it a name.

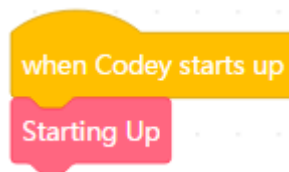


- Then, the **Define** block for the function will appear in the Scripts area.

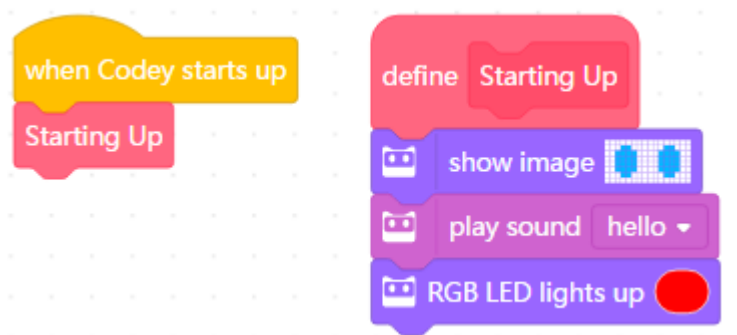


- What programs do you want to run each time Codey Rocky starts up? Design programs under the “**Define start up**” block.

- After they finish defining the function, have students attach the “**Start up**” block to the bottom of the event block “**When Codey Rocky starts up**”.



Sample Programs:

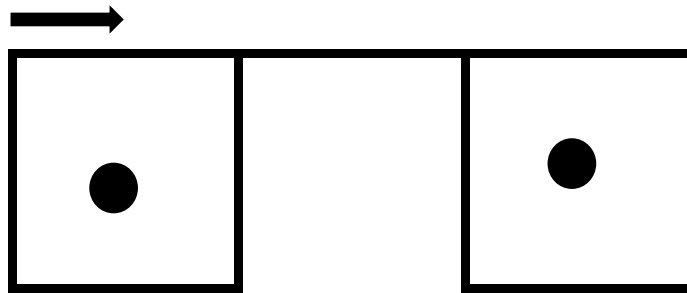


Task 2: Patrol the 1st Floor

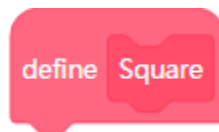
Task 2- Patrol the 1st Floor

Imagine Codey Rocky is a security guard. It has to patrol the passages in the building to keep an eye on properties. Now, it’s patrolling the 1st floor.

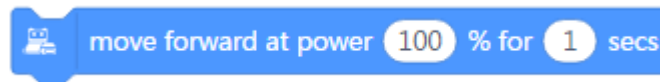
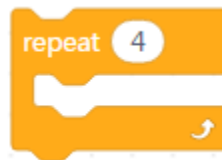
- Design programs to make Codey Rocky patrol the passage by following the black lines as illustrated below.



- Create a function and name it as Square.



- You might need to attach the following blocks to the bottom of the function.



- Under the event block “**When button A pressed**”, call the function Square two times.
- **Challenge:** Add expressions, sounds and lights.

Tips:

1. Before handing out the task cards, teachers need to explain more about the task. Students are likely to miss out details because they seldom enjoy reading texts.

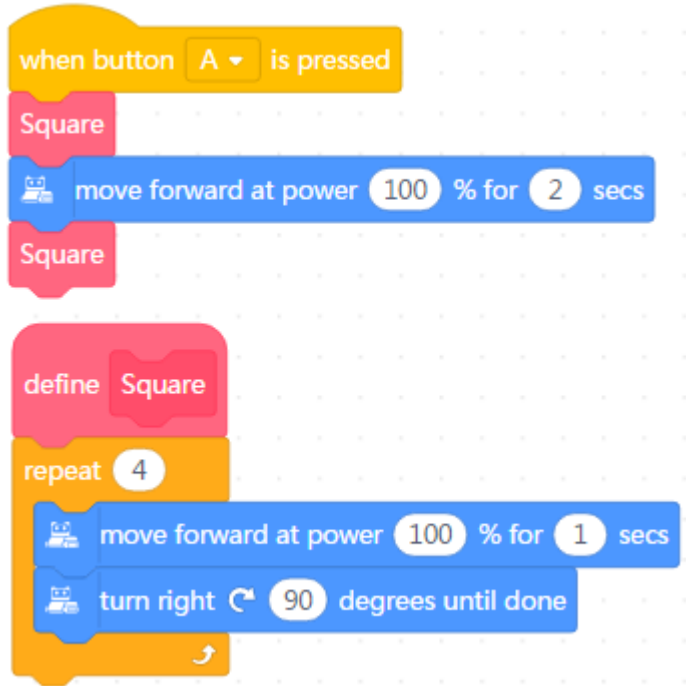
2. There is a chance that Codey Rocky might fail to rotate by accurately 90 degrees. Give Codey Rocky tolerance. If it can patrol in the roughly same route as illustrated in the picture above, then it's fine;

3. The longer distance Codey Rocky walks, the more likely it is to deviate from the original direction. Therefore, it is recommended that you set the walking time to be 1 second or 2 seconds;

4. After Codey Rocky repeats moving forward and turning right four times, it will return to the starting point;

5. Since there are two squares, we should call the square function twice at least in the program.

Sample Programs:



Task 3: Patrol the 2nd Floor

Task 3-Patrol the 2nd Floor

Codey Rocky comes to the 2nd floor. There are more rooms and the passages are more complex than the 1st floor.

Design programs to make Codey Rocky patrol in the black route as illustrated below.

A floor plan diagram consisting of a 2x4 grid of rooms. A thick black line traces a path through the rooms: starting at the top-left corner, moving right, then down, then right, then up, then right, then down, then right, then down, then right, then down. A black arrow points to the start of the path at the top-left corner.

- ❑ You need to create two functions, Upper square, Underneath square.
- ❑ Under the event block “**When button A pressed**”, call the two functions in the program.
Use the **Repeat** block to make your program neat.
- ❑ **Challenge:** Add expressions, sounds and lights.

Tips:

1. Remind students that they should create two functions, like Upper square, Underneath square;
2. There are several ways to make Codey Rocky take the route as shown above. Have students design programs on their own first; then teachers use the following pseudocode to inspire students to complete the task:

Underneath square: move forward 1 sec turn right by 90° <div style="float: right; margin-left: 20px;"> Repeat four times </div>	Upper square: move forward 1 sec turn left by 90° <div style="float: right; margin-left: 20px;"> Repeat four times </div>
When button A pressed: underneath square move forward 1 sec upper square move forward 1 sec <div style="float: right; margin-left: 20px;"> Repeat two times </div>	

Sample Programs:

```

when button A is pressed
  repeat 2
    Underneath Square
    Upper Square
    move forward at power 50 % for 1 secs

define Upper Square
  repeat 4
    move forward at power 50 % for 1 secs
    turn left 90 degrees until done

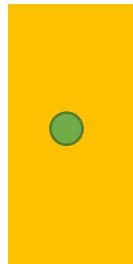
define Underneath Square
  repeat 4
    move forward at power 50 % for 1 secs
    turn right 90 degrees until done
    
```

Task 4: Dancing Turtle

Task 4- Dancing Turtle

We are at a dancing party. Animals are going to show their dance moves one by one. The first one is a little turtle.

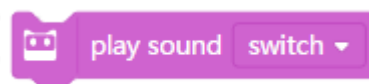
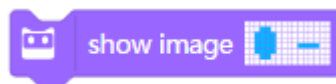
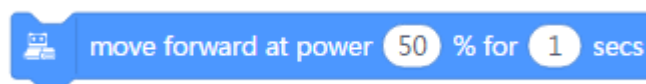
- The little turtle is standing at the center of the rectangle stage, ready to perform his dance moves.:



- Use a function block to compose dance steps for the turtle. Name the function as “**Turtle Dance**”.



- To add dance moves, expressions and sounds, you might need to add the following blocks to the function:



- The turtle is expected to dance on both sides of the stage. In consideration of this, we need to use the function block two times in different places in the program.

Tips:

1. Some students might use the block “move backward“ to make the turtle return to the center of the stage. This can also work.

Sample Programs:

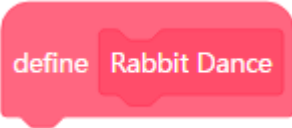
```

when button A is pressed
  show image [Turtle]
  wait 1 secs
  Turtle Dance
  turn left 90 degrees until done
  Turtle Dance
  
```

```

define Turtle Dance
  show image [Turtle]
  play sound hum
  move forward at power 20 % for 2 secs
  turn left 15 degrees until done
  show image [Turtle]
  play sound switch
  turn right 30 degrees until done
  show image [Turtle]
  play sound switch
  turn left 15 degrees until done
  show image [Turtle]
  play sound switch
  turn right 180 degrees until done
  show image [Turtle]
  move forward at power 20 % for 2 secs
  
```

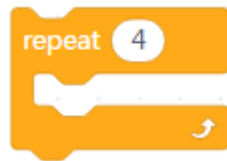
Task 5: Dancing Rabbit

Task 5- Dancing Rabbit	
The second one is a little rabbit.	
<input type="checkbox"/> Use the function block to compose dance steps for the rabbit. Name the function as “ Rabbit Dance ”.	
<input type="checkbox"/> Design dance moves, expressions, and sounds for the rabbit.	

- The rabbit is expected to start from the center of the cruciform stage and perform the dance on four sides. When finishing dancing on one side, the rabbit needs to return to the center of the stage and faces forward another side. As shown below:



- Use the **Repeat** block to make your program neat.



Tips:

1. Some students might call the function in a way that's different from the sample programs.

If students' programs are too complex, show the following sample programs to them.

Sample Programs:

```

when button A is pressed
  repeat 4
    Rabbit Dance
    turn left 90 degrees until done

define Rabbit Dance
  show image [Rabbit]
  play sound [jump]
  move forward at power 50 % for 1 secs
  turn right 15 degrees until done
  show image [Rabbit]
  play sound [low-energy] until done
  turn left 30 degrees until done
  show image [Rabbit]
  play sound [metal-clash] until done
  show image [Rabbit]
  play sound [jump] until done
  turn right 15 degrees until done
  move backward at power 50 % for 1 secs
  
```

Task 6: Dancing Swan

Task 6- Swan Dance

The last one is a swan.

- Create two functions to compose dance moves for the swan.
- Like the rabbit, the swan will start from the center of the cruciform stage and dance on four sides of the stage. Make the swan rotate once at each corner of the square (As shown in the picture on the right).



- You should create two functions. Use one function to represent the dance performances of the swan on each side (Swan Dance). Use the other function to represent the rotations of the swan at each corner (Square Rotate). The second function is used to execute the first function, which forms a nested function. The nested function will make your programs look clean.

```

define Swan Dance
  define Square Rotate
    repeat 4
      wait 1 secs
      turn right 360 degrees until done
      move forward at power 50 % for 1 secs
      turn right 90 degrees until done
  
```

- Add expressions and sounds.

Tips:

1. You need two functions and one of the function is used to execute the other function;
2. The block “wait 1 sec” in the sample programs makes the swan dance steps clearer;
3. Since the nested function is a complicated concept, teachers could help students complete the task by showing them the following pseudocode:

<p>Square Rotate:</p> <pre> wait 1 sec rotate 1 round moveforward turn right by 90° </pre> <p style="text-align: right;">Repeat four times</p>	<p>Swan Dance:</p> <pre> move forward 1 sec square rotate wait 1 sec rotate 180° move forward 1 sec </pre> <p style="text-align: right;">Repeat four times</p>
<p>When button A pressed:</p> <pre> swan rotate turn left by 90° </pre> <p style="text-align: right;">Repeat four times</p>	

Sample Programs:

```

when button A is pressed
  repeat 4
    Swan Dance
    turn left 90 degrees until done

define Square Rotate
  repeat 4
    wait 1 secs
    turn right 360 degrees until done
    move forward at power 50 % for 1 secs
    turn right 90 degrees until done

define Swan Dance
  show image swan
  move forward at power 50 % for 1 secs
  show image swan
  Square Rotate
  wait 1 secs
  show image swan
  move backward at power 50 % for 1 secs
    
```

Self-review Report

Name:

Age:

- Answer the following questions to record your learning outcomes:

Describe what you've learned with one or two sentences.

Describe what you like most and least about this class session briefly.

What I like most

What I like least

Use one or two sentences to articulate why you need to use a function in programs.

How to simplify the following programs using functions? Draw your new programs in the right column.

	
---	--

You can draw out how you feel about this class session at the upper right corner of the self-review report.

<p><u>Function: Oversleep</u> cover head with the quilt; wait for 5 seconds; uncover the quilt; stretch out one hand; turn off the alarm clock.</p>	<p><u>Function: Have a look at the time</u> take up the alarm clock; have a look at the time; put down the clock.</p>	<p><u>Function: Get up</u> stretch yourself; uncover the quilt; sit up.</p>
<p><u>Function: Put on slippers</u> put on the slipper to the left foot; put on the slipper to the right foot; stand up.</p>	<p><u>Function: Walk</u> lift the left foot; stretch forward the left foot and step on the ground; lift the right foot; stretch forward the right foot and step on the ground.</p>	<p><u>Function: Brush teeth</u> take the toothbrush; squeeze the toothpaste; rinse the mouth; brush the teeth all around; rinse the mouth.</p>
<p><u>Function: Wash up</u> turn on the tap; cup the water with hands; wash up with the water; turn off the tap; dry your face with the tow</p> <p style="text-align: right; margin-right: 20px;">Repeat three times</p>	<p><u>Function: Take off the coat</u> If it's a T-shirt, take it off from top to bottom; If the coat has buttons, unbutton it first, then the left arm comes out of the left sleeve, next the right arm comes out of the right sleeve.</p>	<p><u>Function: Take off pants</u> pull down the pants; the left leg comes out of the pants; the right leg comes out of the pants.</p>
<p><u>Function: Take clothes off</u> take off the coat; take off pants.</p>	<p><u>Function: Put on the coat</u> get the left arm into the left sleeve; get the right arm into the right sleeve; if it's a T-shirt, get your head into the shirt; if the coat has buttons, fasten the buttons.</p>	<p><u>Function: Put on pants</u> get the left leg into the pants; get the right leg into the pants.</p>
<p><u>Function: Dress up</u> put on the pants; put on the coat.</p>	<p><u>Function: Carry schoolbag</u> get the right arm through one strap; get the left arm through the other strap; shoulder the schoolbag.</p>	<p><u>Function: Lace the shoes</u> make a circle of the shoelace with the left hand; make a circle of the shoelace with the right hand; tie the two circles</p>
<p><u>Function: Put on shoes</u> get the left feet into the left shoe; get the right feet into the right shoe; lace the left shoe.</p>		

Task 1-Starting up Function

Create a starting up function for Codey Rocky

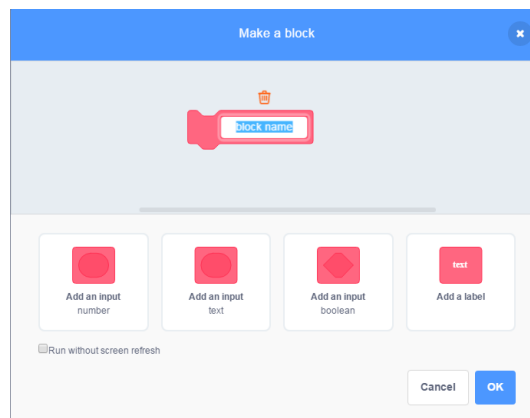
- ❑ Click **My Blocks** at the category bar and select **Make a Block**.

● Variables

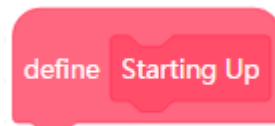
● **My Blocks**

● Infrared

- ❑ Click **Make a Block**. Create a function and give it a name.

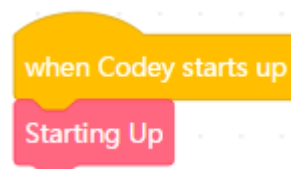


- ❑ Then, the **Define** block for the function will appear in the Scripts area.



- ❑ What programs do you want to run each time Codey Rocky starts up? Design programs under the “**Define start up**” block.

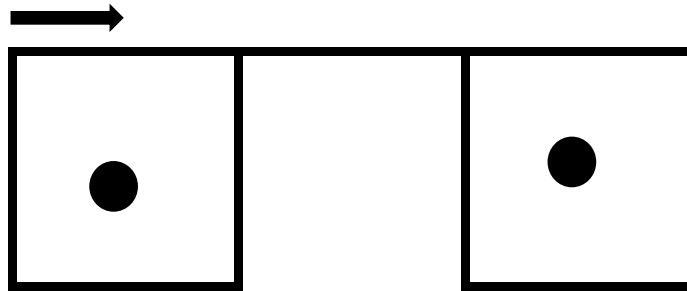
- ❑ After they finish defining the function, have students attach the “**Start up**” block to the bottom of the event block “**When Codey Rocky starts up**”.



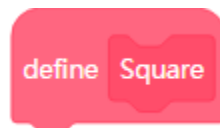
Task 2- Patrol the 1st Floor

Imagine Codey Rocky is a security guard. It has to patrol the passages in the building to keep an eye on properties. Now, it's patrolling the 1st floor.

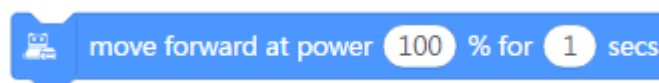
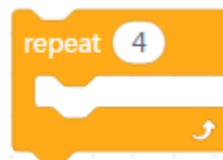
- Design programs to make Codey Rocky patrol the passage by following the black lines as illustrated below.



- Create a function and name it as Square.



- You might need to attach the following blocks to the bottom of the function.

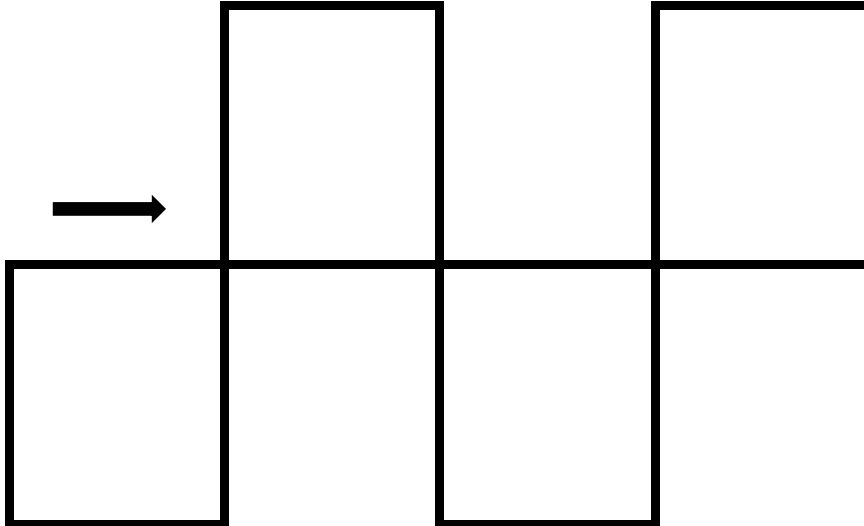


- Under the event block “**When button A pressed**”, call the function Square two times.
- **Challenge:** Add expressions, sounds and lights.

Task 3-Patrol the 2nd Floor

Codey Rocky comes to the 2nd floor. There are more rooms and the passages are more complex than the 1st floor.

- Design programs to make Codey Rocky patrol in the black route as illustrated below.



- You need to create two functions, Upper square, Underneath square.
- Under the event block “**When button A pressed**”, call the two functions in the program.

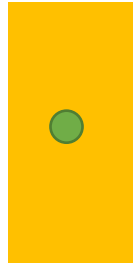
Use the **Repeat** block to make your program neat.

- Challenge:** Add expressions, sounds and lights.

Task 4- Dancing Turtle

We are at a dancing party. Animals are going to show their dance moves one by one. The first one is a little turtle.

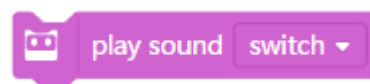
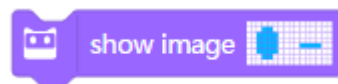
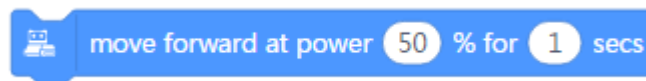
- The little turtle is standing at the center of the rectangle stage, ready to perform his dance moves.:



- Use a function block to compose dance steps for the turtle. Name the function as “**Turtle Dance**”.



- To add dance moves, expressions and sounds, you might need to add the following blocks to the function:



- The turtle is expected to dance on both sides of the stage. In consideration of this, we need to use the function block two times in different places in the program.

Task 5- Dancing Rabbit

The second one is a little rabbit.

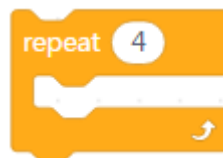
- Use the function block to compose dance steps for the rabbit. Name the function as “**Rabbit Dance**”.



- Design dance moves, expressions, and sounds for the rabbit.
- The rabbit is expected to start from the center of the cruciform stage and perform the dance on four sides. When finishing dancing on one side, the rabbit needs to return to the center of the stage and faces forward another side. As shown below:



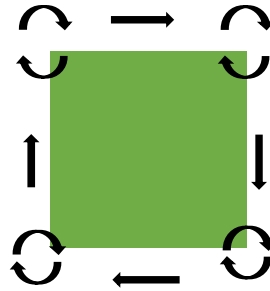
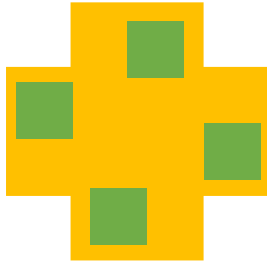
- Use the **Repeat** block to make your program neat.



Task 6- Swan Dance

The last one is a swan.

- Create two functions to compose dance moves for the swan.
- Like the rabbit, the swan will start from the center of the cruciform stage and dance on four sides of the stage. Make the swan rotate once at each corner of the square (As shown in the picture on the right).



- You should create two functions. Use one function to represent the dance performances of the swan on each side (Swan Dance). Use the other function to represent the rotations of the swan at each corner (Square Rotate). The second function is used to execute the first function, which forms a nested function. The nested function will make your programs look clean.

```

define Swan Dance
  define Square Rotate
    repeat 4
      wait 1 secs
      turn right 360 degrees until done
      move forward at power 50 % for 1 secs
      turn right 90 degrees until done
  
```

- Add expressions and sounds.